Unicenter

Management Services
MODS
Programming and Administration Guide

4th edition

P01- 360

Computer Associates
The Software That Manages eBusiness

Booklist

Manual History

Edition	Publication Number	Min. MS Level	Publish Date
1st Edition	P01-30360-00	ASM 3.0	July 1993
2nd Edition	P01-360-31-00	MS 3.1	May 1994
3rd Edition	P01-360	MS 3.2	March 1995
4th Edition	P01-360	MS 3.2	May 1999
4th Edition, Update 1	P01-360	MS 3.7	August 1999
4th Edition, Update 2	P01-360	MS 4.0	March 2000

New editions incorporate any updates issued since the previous edition.

SOLVE Management Services™ Managed Object Development Services Programming and Administration Guide, 4th Edition,Update2, P01-360

Table of Contents

What's New in This Edition xxvii

About This Manual xxix

How This Book Is Organized xxix
Assumed Knowledge xxx
Notational Conventions xxx
Related Documentation xxxi

Part I Introduction

Chapter 1 About MODS 1-1

What is MODS? 1-2

Chapter 2 Concepts and Facilities 2-1

MODS Facilities 2-1
Object Services 2-3
The Application Register 2-3
Panel Maintenance 2-3

Common Application Services (CAS) 2-4
CAS Programming Interface 2-5
Menus 2-6
Lists 2-6
Panel Domains 2-8
Text Panels 2-8
Help 2-8
The Help Hierarchy 2-9
Messages 2-10
Validation 2-11
Tables 2-11
Prompted Fields 2-11
Other Types of Validation 2-12
Criteria 2-12
Commands 2-12
Text Editor 2-13
Text Browse Facility 2-13
Function Key Areas 2-13
Report Maintenance 2-14
Mapping Services 2-15
What is Mapping Services? 2-15
The Need for Mapping Services 2-15
Maps and Mapped Data Objects 2-16
The SOLVE Map Library 2-16
MODS Administration Facilities 2-17
Panel Library Maintenance 2-17
MODS Control File Maintenance 2-17
Control File Maintenance Facilities 2-18
Control File Concatenation 2-18
Concatenation Path 2-18
Merged View 2-18
Automatic Promotion 2-19
Manual Demotion (Lodgement) 2-19
Sharing Control Files 2-19
SOLVE Web File System 2-20

Chapter 3 Getting Around 3-1

Accessing MODS Facilities 3-1
MODS Access Authority 3-5

Part II Maintaining Application Components

Chapter 4 Registering an Application 4-1

About Application Definitions 4-1

Defining an Application Definition 4-2
- The Application Definition Menu 4-2
- Adding an Application Definition 4-3

Maintaining Application Definitions 4-4
- Browsing an Application Definition 4-4
- Updating an Application Definition 4-4
- Deleting an Application Definition 4-4
- Copying an Application Definition 4-5
- Listing Application Definitions 4-5

Maintaining Application Groups 4-7

Chapter 5 Naming Standards 5-1

Naming Standards 5-1
- Application Definitions 5-2
- Panels 5-2
- NCL Procedures 5-2
- CAS Components 5-2
- Object Class Specifications 5-2
- Reports 5-2
- Map Definitions 5-3
- Externally Visible Entities 5-3

Chapter 6 Maintaining Panels 6-1

About Panel Maintenance 6-1
- Concepts and Terminology 6-2
  - Retrieving Panels From Panel Libraries 6-2

Defining and Maintaining Panels 6-3
- The MODS : Panel Maintenance Menu 6-3
  - Library Selection List 6-5
  - Selecting a Panel Maintenance Function 6-6
- Adding a Panel Definition 6-6
- Listing Panel Definitions 6-7
  - Special List Commands 6-10
- Browsing a Panel Definition 6-10

Updating a Panel Definition 6-10
Deleting a Panel Definition 6-11
Copying a Panel Definition 6-11
Viewing a Panel Definition in Display Format 6-11
Printing a Panel Definition 6-11
Renaming a Panel Definition 6-12
Display Information About a Panel Definition 6-12
Moving and Copying Panel Definitions Between Libraries 6-13
Panel Move/Copy List 6-15
Searching Panels for a Character String 6-16

Designing Panels 6-17
Panel Control Statements 6-17
Data in Panels 6-18
Panel Design 6-19
Field Characters 6-19
Field Types 6-20
Allowing Long Field Names in Short Fields 6-22
Output Padding and Justification 6-22
Field Level Justification 6-23
Variable Level Justification 6-23
Input Padding and Justification 6-25
Displaying Function Key Prompts 6-25

#ALIAS 6-27

#ERR 6-29

#FLD 6-32

#NOTE 6-47

#OPT 6-48

#TRAILER 6-55

Chapter 7 Maintaining Menus 7-1

About Menu Definitions 7-1

Defining a Menu 7-2
The Menu Definition Maintenance Menu 7-2
Adding a Menu Definition 7-3

Maintaining Menu Definitions 7-8
Listing Menu Definitions 7-8
Browsing a Menu Definition 7-10
Updating a Menu Definition 7-10
Deleting a Menu Definition 7-11

Copying a Menu Definition 7-11
Viewing a Menu Definition 7-11

Chapter 8 Maintaining Lists 8-1

About Lists..... 8-1
List Description Panel..... 8-2
List Criteria Panel 8-3
List Entry Line Presentation Attributes..... 8-3
List Format Panel 8-3
List Entry Line Fields Panel..... 8-4
List Cache..... 8-4

Defining a List 8-4
List Definition Menu..... 8-5
Adding a List Definition 8-6

Maintaining List Definitions..... 8-15
Listing List Definitions 8-15
Browsing a List Definition..... 8-19
Updating a List Definition 8-20
Deleting a List Definition..... 8-20
Copying a List Definition..... 8-20
Resetting the List Cache..... 8-20

Chapter 9 Maintaining Panel Domains..... 9-1

About Panel Domains 9-1
Text Panel Element 9-2
Panel Path Definitions..... 9-2
Panel Domains 9-2

Defining a Panel Domain..... 9-3
Panel Domain Definition Menu 9-3
Adding a Panel Domain Definition..... 9-4

Maintaining Panel Domain Definitions 9-5
Listing Panel Domain Definitions..... 9-5
Browsing a Panel Domain Definition 9-7
Updating a Panel Domain Definition..... 9-8
Copying a Panel Domain Definition 9-8
Deleting a Panel Domain Definition 9-8
Reloading Panel Domains..... 9-9

Maintaining Element Definitions 9-9
Listing Element Definitions 9-9
Adding an Element Definition 9-12
Browsing an Element Definition 9-15
Updating an Element Definition 9-15
Deleting an Element Definition 9-15
Copying an Element Definition 9-15
Viewing an Element Definition 9-15
Editing an Element Definition 9-16

Maintaining Path Definitions 9-16
Listing Path Definitions 9-16
Adding a Path Definition 9-18
Browsing a Path Definition 9-20
Updating a Path Definition 9-20
Deleting a Path Definition 9-20
Copying a Path Definition 9-20

Chapter 10 Maintaining Help 10-1

About Help 10-1

Maintaining Application Level Help 10-4
Adding an Application Level Help File 10-5
Browsing an Application Level Help File 10-6
Updating an Application Level Help File 10-7
Deleting an Application Level Help File 10-7
Copying an Application Level Help File 10-7
Listing Application Level Help Files 10-9
Printing an Application Level Help File 10-11
Printing an Application Level Help File in View Format 10-11
Viewing an Application Level Help File 10-11

Maintaining Function Level Help 10-12
Adding a Function Level Help File 10-14
Browsing a Function Level Help File 10-14
Updating a Function Level Help File 10-15
Deleting a Function Level Help File 10-15
Copying a Function Level Help File 10-15
Printing a Function Level Help File 10-15
Printing a Function Level Help File in View Format 10-16
Viewing a Function Level Help File 10-16
Listing Function Level Help Files 10-16

Maintaining Window Level Help 10-17
Adding a Window Level Help File 10-18
Browsing a Window Level Help File 10-19
Updating a Window Level Help File 10-19
Deleting a Window Level Help File 10-19
Copying a Window Level Help File 10-19
Listing Window Level Help Files 10-20
Printing a Window Level Help File 10-20
Printing a Window Level Help File in View Format 10-20
Viewing a Window Level Help File 10-20
Maintaining Field Level Help 10-21
Adding a Field Level Help File 10-22
Browsing a Field Level Help File 10-22
Updating a Field Level Help File 10-23
Deleting a Field Level Help File 10-23
Copying a Field Level Help File 10-23
Listing Field Level Help Files 10-24
Printing a Field Level Help File 10-24
Printing a Field Level Help File in View Format 10-24
Viewing a Field Level Help File 10-24
Defining an Alias 10-25
Facilities for Help Text Editing and Formatting 10-27
Help Text Editor 10-27
Help Macros 10-27
How to Use This Section 10-28
Display Attributes 10-28
.AT 10-30
.BX 10-32
.CE 10-34
.CH 10-35
.CM 10-36
.CP 10-37
.CT 10-39
.DE 10-41
.LI 10-43
.LN 10-45
.MU 10-46
.NP 10-49

.OP 10-50
.PH 10-52
.RA 10-53
.SH 10-54
.SP 10-55
.TI 10-56

Chapter 11 Maintaining Messages 11-1

About Messages 11-1
Defining Messages 11-1
Message Definition Menu 11-2
Adding a Message Definition 11-3
Maintaining Message Definitions 11-7
Listing Message Definitions 11-7
Browsing a Message Definition 11-9
Updating a Message Definition 11-10
Deleting a Message Definition 11-10
Copying a Message Definition 11-11
Printing a Message Definition 11-11
Viewing a Message Definition 11-11

Chapter 12 Maintaining Tables 12-1

About Tables 12-1
Reloading Validation Tables 12-2
Defining and Maintaining Table Definitions 12-4
Table Maintenance Menu 12-4
Adding a Table Definition 12-6
Browsing a Table Definition 12-9
Updating a Table Definition 12-10
Deleting a Table Definition 12-10
Copying a Table Definition 12-10
Listing Table Definitions 12-10
Listing Table Entries 12-13
Copying a Table Definition and Its Table Entries 12-15
Deleting a Table Definition and Its Table Entries 12-15

Maintaining Table Entries 12-16
Table Entry Definition Menu 12-16
Adding a Table Entry 12-17
Browsing a Table Entry 12-19
Updating a Table Entry 12-19
Deleting a Table Entry 12-19
Copying a Table Entry 12-20
Listing Table Entries 12-20

Chapter 13 Maintaining Criteria 13-1

About Criteria 13-1
Run Time Panel 13-2
Criteria Exit 13-2
Data Source 13-2
Exit Parameters 13-2
Substitution of Variable Data 13-3

Defining Criteria 13-3
Criteria Definition Menu 13-3
Adding a Criteria Definition 13-4

Maintaining Criteria Definitions 13-8
Listing Criteria Definitions 13-8
Browsing a Criteria Definition 13-11
Updating a Criteria Definition 13-11
Deleting a Criteria Definition 13-12
Copying a Criteria Definition 13-12

Chapter 14 Maintaining Commands 14-1

About Commands 14-1

Defining a Command 14-1
Command Definition Menu 14-2
Adding a Command Definition 14-3

Maintaining Command Definitions 14-5
Listing Command Definitions 14-5
Browsing a Command Definition 14-8
Updating a Command Definition 14-8
Deleting a Command Definition 14-8
Copying a Command Definition 14-9
Reloading Command Definitions 14-9

Chapter 15 Printing MODS Components 15-1

About MODS Component Reports 15-1

Printing Components Using the REP Option 15-2
Identifying Definition Variables 15-5

Part III MODS Administration

Chapter 16 Maintaining Panel Libraries 16-1

About Panel Libraries 16-1

Maintaining Panel Libraries 16-2
Accessing the Panel Library Maintenance Facilities 16-2
Copying Panels Between Libraries (Any Path) 16-3
Selecting Panels to Copy 16-4

Maintaining Library Definitions 16-6

Chapter 17 Maintaining the MODS Control File 17-1

About Control Files 17-2
Accessing the Control File Maintenance Facilities 17-4
Applying APAR and PUT Tapes 17-6

Copying Components Between Control Files 17-8
Selecting Application Entities to Copy 17-10
Selecting Common Entities to Copy 17-13
Resetting Selections 17-14
Copying Selected Data 17-14
Accessing the Log 17-16
Browsing Log Records 17-16
Printing Log Records 17-17
Clearing Log Records 17-18

Moving Components Between Control Files 17-18
Selecting Application Entities to Move 17-21
Selecting Common Entities to Move 17-21
Resetting Selections 17-21
Moving Selected Data 17-22
Accessing the Log 17-23

Deleting Components from the Control File 17-23
Selecting Application Entities to Delete 17-25

Selecting Common Entities to Delete 17-25
Resetting Selections 17-25
Deleting Selected Data 17-25
Accessing the Log 17-26
Comparing Control Files 17-26
Selecting Application Entities to Compare 17-30
Selecting Common Entities to Compare 17-30
Resetting Selections 17-30
Comparing Selected Data 17-30
Accessing the Log 17-32
Applying Differences 17-32
Applying the Differences 17-34
Accessing the Log 17-35
Browsing the Control File 17-36
Browsing Application Entities 17-37
Browsing Common Entities 17-38
Searching the Control File 17-38
Considerations 17-39
Listing the Control File 17-40
Resetting the MODS Cache 17-42

Part IV CAS Programming Guide

Chapter 18 CAS Programming Interface ($CACALL) 18-1

About $CACALL 18-1
The $CACALL API 18-3
Action : BUILD
Class : CFPATH 18-5
Action : BUILD
Class : CRITERIA 18-7
Action : BUILD
Class : FKA 18-10
Action : BUILD
Class : IDTEXT 18-13
Action : BUILD
Class : MESSAGE 18-15

Action : DISPLAY
Class : DATA 18-17

Action : DISPLAY
Class : HELP..... 18-21

Action : DISPLAY
Class : LIST 18-23

Action : DISPLAY
Class : MENU..... 18-26

Action : DISPLAY
Class : MESSAGE..... 18-28

Action : EDIT
Class : DATA 18-29

Action : EXECUTE
Class : COMMAND 18-33

Action : LOAD
Class : COMMAND 18-36

Action : GET
CLASS : TENTRY 18-37

Action : LOAD
Class : PDOMAIN..... 18-39

Action : LOAD
Class : TABLE..... 18-41

Action : NAVIGATE
Class : PDOMAIN..... 18-43

Action : VALIDATE
Class : DATA 18-47

Chapter 19 Menu Service Procedure Interface 19-1

About Menu Service Procedure Interface..... 19-1

$MHOPT=ENTRY..... 19-3

&$MHOPT=SELECT 19-5

&$MHOPT=RETURN..... 19-7

$MHOPT=EXIT..... 19-8

&$MHOPT=COMMAND..... 19-9

&$MHOPT=TIMEOUT..... 19-10

Chapter 20 List Service Procedure Interface 20-1

About the List Service Procedure Interface 20-1
$LHOPT=ACTION 20-3
&$LHOPT=ADD 20-6
&$LHOPT=ALL 20-9
&$LHOPT=COMMAND 20-12
&$LHOPT=GET 20-15
&$LHOPT=INIT 20-19
&$LHOPT=TERM 20-22

Chapter 21 List Exit Procedure Interface..... 21-1

About the List Exit Procedure Interface 21-1
$LHOPT=INIT 21-3
&$LHOPT=ENTRY 21-6
$LHOPT=TERM 21-8

Chapter 22 Criteria Exit Procedure Interface 22-1

About Criteria Exit Procedure Interface 22-1
&$CROPT=INIT 22-3
&$CROPT=TERM 22-5

Chapter 23 Table Entry Validation Exit Procedure Interface 23-1

About Table Entry Validation Exit Procedure Interface 23-1
&$VMEXFUNC=ADD 23-3
&$VMEXFUNC=DELETE 23-5
&$VMEXFUNC=UPDATE 23-7

Part V Mapping Services

Chapter 24 Mapping Services 24-1

About Mapping Services 24-1
- Abstract Syntax Notation One (ASN.1) 24-1
- ASN.1 Type Assignments 24-2
- Defining the Logical Structure of Data 24-2
- Referencing Logical Data Structures From NCL 24-4
- Defining the Physical Structure of Data 24-4
  - Component Tags 24-5
  - Local Form 24-5
- Data Interchange Between Open Systems 24-5

Map Source Definitions 24-6
- The ASN.1 Language 24-6
- Maps and ASN.1 Modules 24-6
  - Mapping Services Considerations 24-6
- ASN.1 Module Layout 24-7
  - ASN.1 Comments 24-7
  - Module Identifier 24-8
  - Module Definitions 24-8
  - Exports 24-8
  - Imports 24-8
  - Type Assignments 24-8
  - Value Assignments 24-9
- The SOLVE ASN.1 Compiler's Interpretation of the ASN.1 Module 24-9
  - Use of Comments 24-9
  - Module Identifier 24-9
  - Default Tagging 24-9
  - Exports 24-9
  - Imports 24-10
  - Type Assignments 24-10
  - Value Assignments 24-10
- Map Components 24-10
  - Component Definition 24-10
  - The Mapped Data Object as a Component 24-11
  - Component Name Hierarchy 24-11
  - Map Closure 24-11
- Data Tagging 24-12
  - ASN.1 Tags 24-12
  - Explicit and Implicit Tagging 24-13

MDO Tags 24-14
Mapping Directives 24-14
MDO Tag Generation 24-14
Local Form Control 24-16
Default Formatting 24-17
Local Form for INTEGER 24-18
External Form for REAL 24-18
Type Description and Formats 24-18
ASN.1 Types 24-18
NCL Reference, Type Checking, and Data Behavior 24-19
The SET Type 24-20
The SET OF Type 24-21
The SEQUENCE Type 24-22
The SEQUENCE OF Type 24-23
The CHOICE Type 24-24
The BOOLEAN Type 24-25
The INTEGER Type 24-25
The BIT STRING Type 24-27
The OCTET STRING Type 24-29
The HEX STRING Type 24-30
The NULL Type 24-31
The OBJECT IDENTIFIER Type 24-31
The ObjectDescriptor Type 24-32
The EXTERNAL Type 24-32
The REAL Type 24-32
The ENUMERATED Type 24-34
The NumericString Type 24-35
The PrintableString Type 24-35
The TeletexString Type 24-36
The VideotexString Type 24-36
The IA5String Type 24-37
The UTCTime Type 24-37
The GeneralizedTime Type 24-38
The GraphicString Type 24-38
The VisibleString Type 24-39
The GeneralString Type 24-40
The ANY and ADB Types 24-40

Chapter 25 Maintaining Maps 25-1

About Maps 25-1
The SOLVE Map Library Structure 25-1
Creating and Maintaining the Map Source 25-2
Compiling a Map 25-2
Loading a Map 25-3

Defining a Map 25-4
Mapping Services Primary Menu 25-4
Adding a Map Definition 25-5

Maintaining Map Definitions 25-6
Listing Map Definitions 25-6
Browsing a Map Definition 25-10
Browsing the ASN.1 Source Code for a Map 25-10
Updating a Map Definition 25-10
Deleting a Map Definition 25-11
Copying a Map Definition 25-11
Printing a Map Definition and its ASN.1 Source Code 25-11
Compiling a Map's ASN.1 Source Code 25-11
Viewing a Map Structure 25-12

Part VI Reference Material

Appendix A Customer Support Services A-1

Technical Support A-1

Sterling Software Web Site A-2

Local Support Centers A-2
Europe A-3
North America A-5
South America A-5
Pacific Rim A-5

User Conferences A-6

Appendix B Text Editor Commands B-1

Using the Text Editor Commands B-1

Line Commands B-1

Primary Commands B-6

Appendix C Panel Editor C-1

Adding a Panel Definition C-1
Terminating a Panel Edit C-3
Scrolling Edit Data C-3
Format of Edit Panel C-4
Homing the Cursor (F12/F24) C-4

Panel Editor Primary Commands C-4

Panel Editor Line Commands C-12
Identifying Blocks of Lines C-13

Appendix D Shorthand Time and Date Formats D-1

Appendix E List Panel Attributes E-1

Appendix F SOLVE Web File Utilities F-1

About SOLVE Web File Utilities F-2

Glossary

Index

Figures and Tables

Figure 2-1.	MODS Application Components	2-2
Figure 2-2.	CAS Components Definition and Run-Time Interaction	2-5
Figure 2-3.	The Help Hierarchy	2-9
Figure 2-4.	Sample Panel Showing Help Windows	2-10
Figure 3-1.	MODS Facilities Menu Structure	3-2
Figure 3-2.	MODS : Primary Menu	3-3
Figure 3-3.	CAS : Maintenance Menu	3-4
Figure 3-4.	UAMS : Access Authorities Panel	3-5
Figure 3-5.	UAMS : MODS Details Panel	3-6
Figure 4-1.	MODS : Application Definition Menu	4-2
Figure 4-2.	MODS : Application Definition Panel	4-3
Figure 4-3.	MODS : Application Definition List (page 1)	4-5
Figure 4-4.	MODS : Application Definition List (page 2)	4-6
Figure 4-5.	Sample Group Table Definition	4-7
Figure 6-1.	MODS : Panel Maintenance Menu	6-4
Figure 6-2.	MODS : Panel Library List	6-5
Figure 6-3.	MODS : Panel List	6-7
Figure 6-4.	MODS : Copy Panel	6-9
Figure 6-5.	MODS : Panel Information Panel	6-12
Figure 6-6.	MODS : Panel Move/Copy Menu	6-14
Figure 6-7.	MODS : Panel Move/Copy List	6-15
Figure 6-8.	MODS : Search Panel Definitions Panel	6-16
Figure 6-9.	Sample Panel Using Default Field Characters	6-21
Figure 6-10.	Sample Panel Display	6-22
Figure 7-1.	CAS : Menu Definition Menu	7-2
Figure 7-2.	CAS : Menu Description Panel	7-3

Figure 7-3. CAS : Menu Options Panel 7-5
Figure 7-4. CAS : Menu Input Fields Panel 7-7
Figure 7-5. CAS : Menu Definition List Panel (screen 1) 7-9
Figure 7-6. CAS : Menu Definition List Panel (screen 2) 7-9
Figure 7-7. Viewing a Menu That You Have Defined in Display Format. . 7-11
Figure 8-1. CAS : List Definition Menu 8-5
Figure 8-2. CAS : List Description Panel 8-6
Figure 8-3. CAS : List Criteria Panel 8-10
Figure 8-4. CAS : List Entry Line Presentation Attributes 8-12
Figure 8-5. CAS : List Format Panel 8-13
Figure 8-6. CAS : List Entry Line Fields Panel 8-15
Figure 8-7. CAS : List Definition List (page 1) 8-16
Figure 8-8. CAS : List Definition List (page 2) 8-16
Figure 8-9. CAS : List Definition List (page 3) 8-17
Figure 8-10. CAS : List Definition List (page 4) 8-17
Figure 8-11. CAS : List Definition List (page 5) 8-18
Figure 9-1. CAS : Panel Domain Definition Menu 9-3
Figure 9-2. CAS : Panel Domain Definition Panel 9-4
Figure 9-3. CAS : Panel Domain Definition List (page 1) 9-6
Figure 9-4. CAS : Panel Domain Definition List (page 2) 9-6
Figure 9-5. CAS : Panel Domain Element Definition List (page 1) 9-10
Figure 9-6. CAS : Panel Domain Element Definition List (page 2) 9-10
Figure 9-7. CAS : Panel Domain Element Definition List (page 3) 9-11
Figure 9-8. CAS : Panel Domain Element Definition Panel 9-13
Figure 9-9. CAS : Panel Domain Path Definition List (page 1) 9-17
Figure 9-10. CAS : Panel Domain Path Definition List (page 2) 9-17
Figure 9-11. CAS : Panel Domain Path Definition Panel 9-19
Figure 10-1. Help Maintenance Panel Navigation 10-3
Figure 10-2. CAS : Application Level Help Definition Menu 10-4
Figure 10-3. CAS : Application Level Help Definition Panel 10-5
Figure 10-4. CAS : Application Level Help File 10-6
Figure 10-5. CAS : Copy Help Panel 10-8
Figure 10-6. CAS : Application Level Help Definition List (page 1) 10-9
Figure 10-7. CAS : Application Level Help Definition List (page 2) 10-10
Figure 10-8. Application Level Help File in Display Format 10-12
Figure 10-9. CAS : Function Level Help Definition Menu 10-13
Figure 10-10. CAS : Window Level Help Definition Menu 10-17
Figure 10-11. CAS : Field Level Help Definition Menu 10-21
Figure 10-12. CAS : Select Alias Help Panel 10-25
Figure 10-13. CAS : Select Alias Help List 10-26
Figure 10-14. CAS: Untitled Help 10-47
Figure 11-1. CAS : Message Definition Menu 11-2
Figure 11-2. CAS : Message Text/Explanation Panel 11-3

Figure 11-3. CAS : Message System/User Action Panel 11-5
Figure 11-4. CAS : Message Definition List (page 1)..... 11-8
Figure 11-5. CAS : Message Definition List (page 2)..... 11-8
Figure 11-6. CAS : Message Information Panel..... 11-10
Figure 12-1. CAS Table Maintenance Panel Navigation 12-3
Figure 12-2. CAS : Table Maintenance Menu..... 12-4
Figure 12-3. CAS : Table Definition Menu..... 12-5
Figure 12-4. CAS : Table Description Panel..... 12-6
Figure 12-5. CAS : Table Text Fields Panel..... 12-9
Figure 12-6. CAS : Table Definition List (page 1)..... 12-11
Figure 12-7. CAS : Table Definition List (page 2)..... 12-11
Figure 12-8. CAS : Table Definition List (page 3)..... 12-12
Figure 12-9. CAS : Table Entry Definition List (page 1)..... 12-14
Figure 12-10. CAS : Table Entry Definition List (page 2)..... 12-14
Figure 12-11. CAS : Table Entry Definition List (page 3)..... 12-15
Figure 12-12. CAS : Table Entry Definition Menu..... 12-16
Figure 12-13. CAS : Table Entry Definition Panel 12-17
Figure 12-14. CAS : Table Entry Definition List (page 1)..... 12-20
Figure 12-15. CAS : Table Entry Definition List (page 2)..... 12-21
Figure 12-16. CAS : Table Entry Definition List (page 3)..... 12-21
Figure 13-1. CAS : Criteria Definition Menu..... 13-3
Figure 13-2. CAS : Criteria Description Panel..... 13-5
Figure 13-3. CAS : Criteria Text Panel 13-7
Figure 13-4. CAS : Criteria Exit Parameters panel 13-8
Figure 13-5. CAS : Criteria Definition List (page 1)..... 13-9
Figure 13-6. CAS : Criteria Definition List (page 2)..... 13-9
Figure 13-7. CAS : Criteria Definition List (page 3)..... 13-10
Figure 14-1. CAS : Command Definition Menu 14-2
Figure 14-2. CAS : Command Definition Panel..... 14-3
Figure 14-3. CAS : Command Definition List (page 1) 14-6
Figure 14-4. CAS : Command Definition List (page 2) 14-6
Figure 14-5. CAS : Command Definition List (page 3) 14-7
Figure 15-1. Report Writer : Report List..... 15-3
Figure 15-2. PSM : Confirm Printer..... 15-3
Figure 15-3. MODS : Definition Report Search..... 15-4
Figure 15-4. Report Writer : Report List..... 15-5
Figure 15-5. Report Writer : Report Information 15-6
Figure 15-6. CAS : Table Entry Definition Menu..... 15-6
Figure 15-7. CAS : Table Entry Definition List..... 15-7
Figure 16-1. MODS : Panel Library Maintenance Menu 16-2
Figure 16-2. MODS : Panel Copy List..... 16-4
Figure 16-3. MODS : Library Definition Menu 16-6
Figure 17-1. MODS : Definition Utility Menu..... 17-5

Figure 17-2. Control File Maintenance—Panel Navigation 17-7
Figure 17-3. MODS : Copy File Specification panel 17-8
Figure 17-4. MODS : Copy Definition Menu 17-9
Figure 17-5. MODS : Application List 17-11
Figure 17-6. MODS : Component List (for Application Data) 17-12
Figure 17-7. MODS : Component List (for Common Data) 17-13
Figure 17-8. MODS : Control File Entity Copy Panel 17-15
Figure 17-9. MODS : Log Menu 17-16
Figure 17-10. MODS : Log Records Panel 17-17
Figure 17-11. PSM : Confirm Printer panel 17-17
Figure 17-12. MODS : Move File Specification Panel 17-19
Figure 17-13. MODS : Move Definition Menu 17-20
Figure 17-14. MODS : Control File Entity Move panel 17-22
Figure 17-15. MODS : Delete File Specification Panel 17-23
Figure 17-16. MODS : Delete Definition Menu 17-24
Figure 17-17. MODS : Control File Entity Delete Panel 17-26
Figure 17-18. MODS : Compare File Specification Panel 17-27
Figure 17-19. MODS : Compare Definitions Menu 17-29
Figure 17-20. MODS : Control File Entity Compare Panel 17-31
Figure 17-21. MODS : Apply File Specification Panel 17-32
Figure 17-22. MODS : Apply Definition Menu 17-33
Figure 17-23. MODS : Control File Entity Apply Panel 17-35
Figure 17-24. MODS : Browse File Specification Panel 17-36
Figure 17-25. MODS : Browse Definition Menu 17-37
Figure 17-26. MODS : Search Definition Panel 17-38
Figure 17-27. MODS : Unformatted List of File Specification Panel 17-40
Figure 17-28. MODS : Unformatted Records List 17-41
Figure 25-1. Registering and Compiling a Map 25-3
Figure 25-2. Mapping Services : Primary Menu 25-4
Figure 25-3. Mapping Services : Map Definition Panel 25-5
Figure 25-4. Mapping Services : Map Definitions List (page 1) 25-7
Figure 25-5. Mapping Services : Map Definitions List (page 2) 25-7
Figure 25-6. Mapping Services : Map Definitions List (page 3) 25-8
Figure 25-7. Mapping Services : Map Definitions List (page 4) 25-8
Figure 25-8. Browsing ASN.1 Source Code for a Map. 25-10
Figure 25-9. Mapping Services : Compiler Messages. 25-12
Figure 25-10. Mapping Services : View Map Structure Panel. 25-12
Figure B-1. Adding Lines in the Text Editor - part 1 ..... B-5
Figure B-2. Adding Lines in the Text Editor - part 2 ..... B-6
Figure B-3. Other Line Commands in the Text Editor B-6
Figure C-1. MODS : Edit Panel C-2
Figure C-2. COPY Command Example C-7
Figure C-3. CREATE Command Example C-8

Figure C-4.	Data After CREATE Command Has Deleted 10 Lines	C-9
Figure C-5.	Example of a Block of Lines Being Copied	C-13
Figure C-6.	Example of an O Line Command	C-15
Figure C-7.	Example of a Completed O Line Command	C-15
Figure F-1.	CAS : Maintenance Menu	F-2
Figure F-2.	Web File Utilities : Top Level Directory Summary	F-2
Figure F-3.	Web File Utilities : Directory Listing	F-3
Table 10-1.	Help Macros	10-27
Table 14-1.	Actions Supported by CAS	14-4
Table 17-1.	MODS Components and Subcomponents/Entities	17-3
Table C-1.	Scroll Amounts	C-3
Table C-2.	Panel Editor Primary Command Summary	C-5
Table C-3.	Panel Editor Line Command Summary	C-12
Table D-1.	Shorthand Time Formats (HH.MM)	D-1
Table D-2.	Shorthand Time Formats (HH.MM.SS)	D-2
Table D-3.	Shorthand Date Formats	D-3
Table E-1.	Panel Attribute Values	E-1

What's New in This Edition

This is the 2nd update of the 4th edition of the *Managed Object Development Services Programming and Administration Guide* (publication number P01-360).

Since the last update of this manual, Chapter 18, *CAS Programming Interface ($CACALL)*, has been updated for the VALIDATE action. The following new values have been added to the EDITS parameter:

- 22 (IP address)
- 23 (Time)
- 24 (Time)

About This Manual

This manual is for System Administrators and System Programmers. It describes the administration and maintenance facilities of the Managed Object Development Services (MODS), and also provides a MODS programming guide.

How This Book Is Organized

This manual is structured as follows:

- Part I, *Introduction*, contains introductory material and describes the uses of the various MODS components.
- Part II, *Maintaining Application Components*, describes the MODS maintenance facilities, and how to use them to create and tailor:
  - Full-screen panels
  - Common Application Services (CAS) components
- Part III, *MODS Administration*, describes the facilities for maintaining panel libraries and the MODS control file.
- Part IV, *CAS Programming Guide*, is a guide to programming using CAS.
- Part V, *Mapping Services*, is a guide to using Mapping Services in NCL applications. This part of the manual is useful for programmers who require complex data structures in NCL.
- Part VI, *Reference Material*, contains reference material, as well as a glossary and index.

Note

The term *user* as it appears in this manual refers to *end users of the system.*

Assumed Knowledge

This manual assumes that you have a basic familiarity with the SOLVE system. If the SOLVE environment is unfamiliar to you, refer to the *SOLVE Management Services User's Guide* as necessary.

Part IV, *CAS Programming Guide*, and *Part V, Mapping Services*, assume that you have access to the *Network Control Language User's Guide* and the *Network Control Language Reference*.

Notational Conventions

The following conventions apply to command descriptions in this manual:

UPPERCASE Characters

Commands and operands are presented as uppercase characters, but can be entered in upper or lower case.

***Italic* Characters**

Italic characters represent variables and show the type of information, rather than the exact information, that must be supplied. The actual entry replaces the italic description. The types of valid data are described in the *Operands* section of each command.

Underscored Values

An underscored value indicates the default optional value that is assumed for an operand if that operand is not specified.

Braces { }

Braces indicate the available options for a required operand. One of the alternatives listed must be selected. Do not include the braces when entering the desired option.

Square Brackets [ ]

Operands in square brackets, including any accompanying equal signs, are optional. Do not include the square brackets when entering the desired option.

Or-sign |

The or-sign is used to separate options. If a group of options is enclosed by square brackets, and the individual options are separated by or-signs, none of the options in the group has to be chosen. If none of these operands is entered, the default value is used (default values are always underscored).

Commas, Quotes, and Equal Signs

Commas, quotes and equal signs must be entered as shown. When commas and equal signs appear within brackets, they are optional and are to be used only if the accompanying optional operand is used.

Related Documentation

Other documentation useful for MODS administrators includes:

SOLVE Management Services User's Guide
Provides information on the SOLVE environment.

SOLVE:Central Implementation and Administration Guide
This manual describes the Object Services maintenance functions, which form part of the MODS environment.

SOLVE:Central Customization Guide
This manual describes how to use the MODS facilities to customize the SOLVE for systems administration applications.

Additional sources of information for programmers include:

Network Control Language User's Guide
Provides information on using NCL.

Network Control Language Reference
Describes all NCL verbs, built-in functions, and system variables.

Report Writer User's Guide
Describes the report maintenance functions, which form part of the MODS environment, as well as how to use Report Writer in your applications.

Part I

Introduction

1

About MODS

This chapter introduces facilities provided by Managed Object Development Services (MODS). MODS is a development environment that provides you with powerful tools for creating and customising NCL applications.

It provides comprehensive, menu and panel driven, facilities for the definition and maintenance of data, presentation and behavior of applications.

MODS includes the Common Application Services (CAS) functions. These special purpose routines define application elements such as menus and lists that are common to all applications and provide a consistent user interface.

These same facilities allow you to easily customize delivered applications, such as the SOLVE:Central applications, to your installation's requirements.

MODS provides the following benefits:

- Ability to quickly develop NCL based applications through the use of sophisticated, interactive development tools
- High reliability due to the use of common routines for standard application components
- A single interface for accessing data and invoking functions for all applications
- Ability to maintain test and production libraries and utilities to move components from one library to another
- Whole applications or individual components can be easily isolated, modified, or replaced due to the registration of application components

The MODS maintenance facilities have the following features:

- **Panel Driven—**Tailoring is performed through full-screen interactive panels. You do not need a knowledge of NCL to perform tailoring.
- **Data Input Assistance—**Selection lists of valid input are available to assist you in performing tailoring and maintenance functions.
- **Text Editor—**The editor allows easy text entry and alteration. The editor is available for maintaining help text, message text, panels, and presentation formats for lists and reports.
- **Help—**Comprehensive online help is available throughout the MODS maintenance functions. Help is invoked by entering the HELP command or pressing the HELP function key.

What is MODS?

The MODS environment comprises the following:

Application Register

The application register maintains application definitions. All applications that are built using MODS must first be defined in this register; applications are assigned an application identifier which is used to name all components that belong to that application.

Panel Maintenance

A facility for creating and tailoring full-screen panel definitions.

Common Applications Services (CAS)

The Common Application Services (CAS) functions are a collection of high quality special-purpose NCL routines designed to facilitate program development. The CAS application components manage the presentation aspects of applications—such as menus, lists, messages, online help, and panel navigation.

Report Writer

Report Writer is a facility for creating and tailoring report definitions, enabling you to tailor reports to your exact requirements. This facility is documented in the *Report Writer User's Guide*.

Object Services

An object-oriented development environment for defining and maintaining an application's data and methods.

Mapping Services

A facility which enables programmers to define complex data structures for use by NCL applications.

Administration Functions

Facilities for maintaining MODS control libraries (for application component definitions), panel libraries (for panel definitions) and Object Services support functions.

2

Concepts and Facilities

This chapter describes the MODS development environment. MODS provides facilities for defining and maintaining common application components used to build an application.

MODS Facilities

MODS gives you a powerful set of development tools for building your own, or customising delivered, applications.

MODS provides you with facilities for defining, customising, and printing the following application components: panels, menus, lists, reports, help text, messages, commands, criteria and tables.

Programming interfaces are available which enable you to include these components in your own NCL applications.

The Object Services component of MODS provides an object oriented programming environment that is available for customising the SOLVE for systems administration applications.

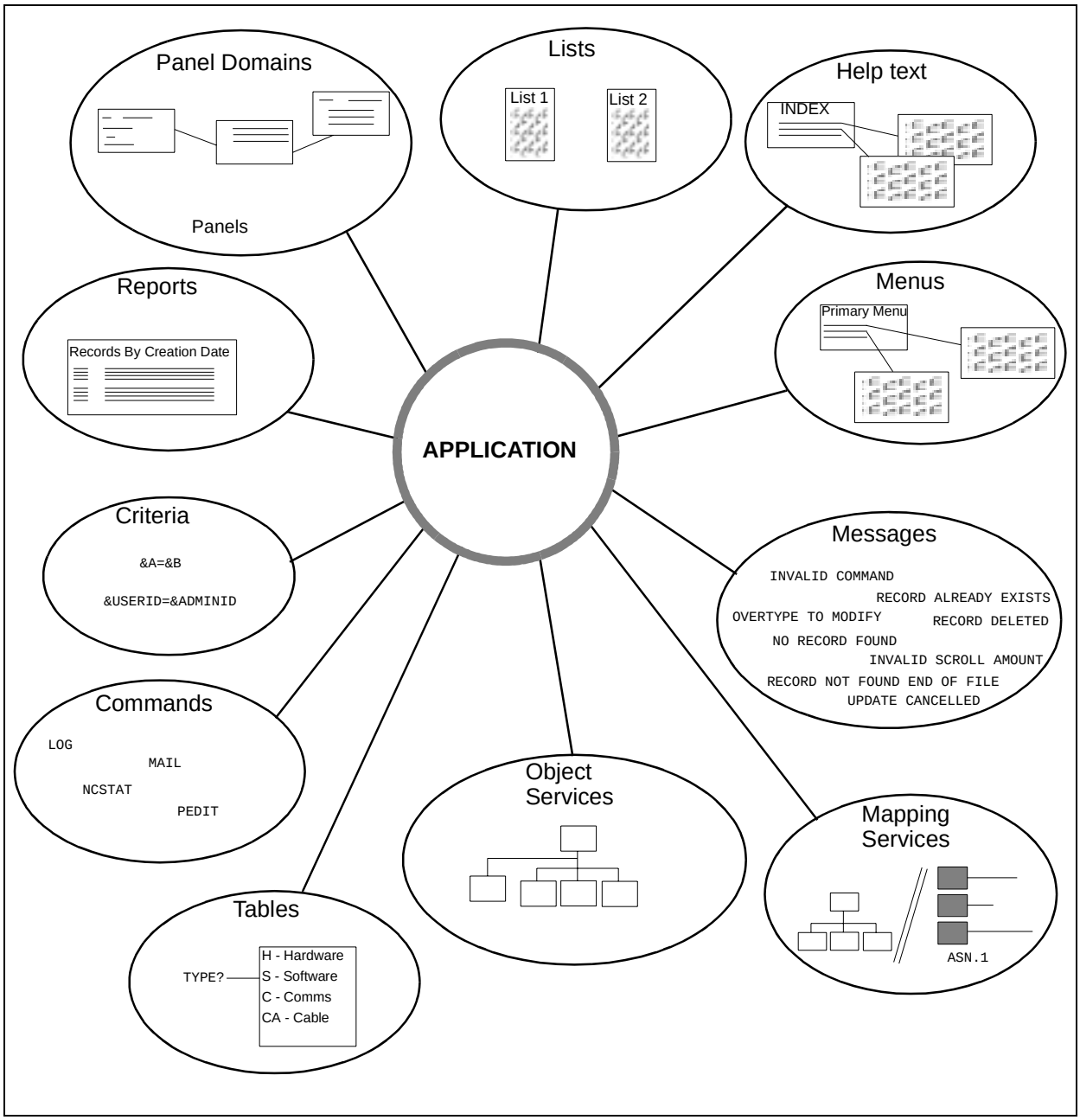
Note

Delivered applications should be customized only if this is expressly stated in the application-specific manuals.

The Mapping Services facility provided by MODS allows you to provide a logical view of data without needing to understand its physical representation.

Figure 2-1 illustrates the components of an NCL application in the MODS environment.

Figure 2-1. MODS Application Components



Object Services

A distinction is made between Object Services based applications and NCL based applications.

Object Services facilities are only used for the SOLVE systems administration applications and are not discussed in this manual.

For details of how to use these facilities, see the *SOLVE:Central Implementation and Administration Guide* and the *SOLVE:Central Customization Guide*.

The Application Register

All applications must be defined in the Application Register.

When you define an application you must specify a unique 3-character identifier (the ***Application ID***), which is used to tag all MODS components belonging to the application.

Facilities for maintaining application definitions are described in Chapter 4, *Registering an Application.*

Chapter 5, *Naming Standards,* contains standards for naming application identifiers and other application components.

Panel Maintenance

The screens used in SOLVE applications are referred to as *panels.* Applications can make use of four general types of panels:

- Menu panels
- List panels
- Data-entry panels
- Text-entry panels

Menu and list panels are defined using the CAS Menu and List maintenance functions and their presentation is controlled by using these facilities. See the sections, *Menus*, on page 2-6 and *Lists*, on page 2-6, for details.

Data entry panels are defined using MODS Panel Maintenance, as described in Chapter 6, *Maintaining Panels*. A panel definition specifies the input and output fields that appear on a panel (as well as some aspects of their behavior), and controls the appearance of the text on the panel (for example, color and highlighting). These panels are invoked using the &PANEL NCL statement. For further details see the *Network Control Language User's Guide*.

Text-entry panels are used to display and maintain freeform text. These panels differ from other panels in that you do not need to provide a panel definition—a standard panel is presented by the CAS text editor facility.

See the section, *Text Editor*, on page 2-13, for further details regarding the text editor.

Data-entry and text-entry panels may can be associated with a *panel domain*. A panel domain represents a collection of associated panels and is used to control the order in which they are displayed. See the section, *Panel Domains*, on page 2-8, for further information.

Common Application Services (CAS)

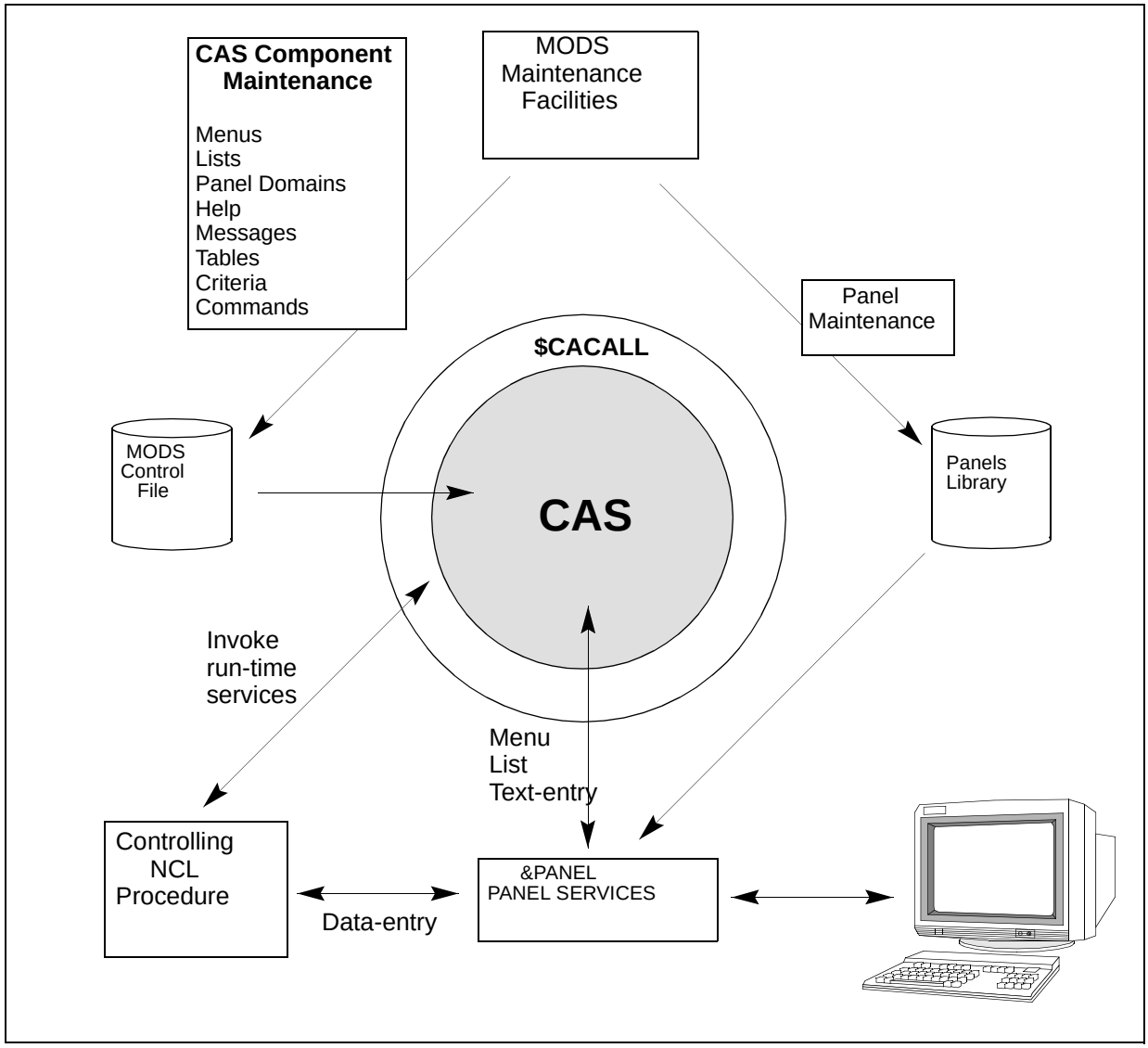
CAS consists of development facilities that are used to define CAS components and run-time services that control interaction with these components during execution.

CAS application components are defined using the MODS maintenance facilities described in this manual. See Part II, *Maintaining Application Components*, for details of how to do this.

NCL based applications must use a controlling NCL procedure to invoke CAS run-time services through the CAS application programming interface (API), $CACALL. The relationship between panels, CAS application component definitions, and CAS run-time services is shown in Figure 2-2.

Object Services provides predefined methods for invoking CAS. The SOLVE:Central applications use these methods.

Figure 2-2. CAS Components Definition and Run-Time Interaction



CAS Programming Interface

The CAS application programming interface ($CACALL) is used to invoke CAS run-time services from a controlling NCL procedure.

The API requires you to specify the action that you want to perform and the type and name of the component to which the action applies.

For example, you can specify a class of MENU with the action DISPLAY and specify the menu name in order to display a menu definition which you have defined using the MODS menu definition facility.

Additional parameters may be required depending on the application component and action being performed.

The syntax for invoking the API is described in Chapter 18, *CAS Programming Interface ($CACALL)*.

Menus

A menu is a panel which presents the user with a number of options. Each option performs a specified action.

A menu is built from a menu definition (see Chapter 7, *Maintaining Menus*) which defines the format of the menu, the menu options and their associated actions, and any input fields required to support an option.

CAS supports panel-skipping between menus. For example, entering **D.C.O.L** goes directly to the last option specified, skipping the display of the three intervening menus.

You can control the behavior of a menu at various processing points, by specifying a menu exit procedure.

When a menu is invoked using the CAS API, CAS builds and displays the menu, and processes the user's selection. When a user selects an option from the menu, and at other defined processing points, CAS calls the menu exit procedure, if it is defined, to perform installation specific processing.

The action that is associated with a menu option can be a further call to the API. For instance, a menu option on a primary menu often leads to a submenu—you can display this menu by specifying a call to the CAS API as the action associated with the menu option. You can invoke other application components (a list, for example) from a menu definition in this way.

Lists

A list displays a series of items from which the user can make a selection of, or perform an action against, one or more items.

A list definition contains identifying information, the name of a service procedure that retrieves the list's entries, the identifier of an (optional) criteria definition that filters items for inclusion in the list, the name of an (optional) exit procedure that performs installation specific processing at various points, and the display format for the list.

The format of a list specifies the placement of list items and static text on the list panel. A list format can cover up to ten screens. That is, if the information you want to display for each item does not fit on a single screen, you can add a second screen, third screen, and so on. The user can scroll between these screens by entering the RIGHT and LEFT commands (or using the appropriate function keys).

The list service procedure retrieves list items and processes requests to perform actions against list items.

The list service procedure checks the list's data source (if it is defined) in order to determine how to retrieve list entries for different sources of data. For example, the data source could be an Object Services class, an INFO/MASTER category, or a filename. This allows multiple list definitions to share the same service procedure. A data source need not be specified—in this case the service procedure retrieves entries from a source defined within the procedure.

A list can contain the identifier of an exit which is an NCL procedure used to perform installation specific processing. The exit procedure is called at various processing points; for example, during list initialization, or after an entry is retrieved. See Chapter 21, *List Exit Procedure Interface*, for details.

The same list definition can be used to build four types of lists:

- **Action lists**—allow the user to apply *actions* (for example, Browse, Update, Delete, Copy) to one or more items on the list.
- **Single Select lists**—allow the user to select one item from the list. The selected item is passed back to the calling procedure.
- **Multiple Select lists**—allow the user to select one or more items from the list. All items selected by the user are passed back to the calling procedure.
- **Numbered lists (Pick lists)**—allow the user to select one item from the list by entering the appropriate number in the **Select Entry** field. The item corresponding to the number that the user selected is then returned to the calling procedure.

You specify the type of list that is to be built from a given list definition in the call to the CAS API.

The CAS list maintenance facilities, described in Chapter 8, *Maintaining Lists*, enable you to define list definitions for use by your applications as well as modifying supplied list definitions.

CAS handles the selection of items for the list and the display of lists to the user, providing: full scrolling functions, the FIND and LOCATE commands, and confirmation of the user's selection.

Panel Domains

A panel domain definition represents a collection of panels and the logical connections between those panels.

Individual panels are specified as domain *elements.* Each element is uniquely named within a domain. Note, however, that one panel can be associated with many elements; this means that a single panel can occur several times within a domain.

A *panel path* is a logical connection between two elements. Panel paths are defined as being from one element to another. Each path has a numeric *weight* associated with it; CAS uses these weights to determine the path to take. You can also associate a condition and/or criteria definition with an element that determines whether it is eligible for display.

Your application calls the CAS API to determine the next panel to be displayed. When you do this you specify the current panel in the domain and the direction in which to move. The API returns the identifier of the next panel to be displayed.

You can also call the API to display a panel showing all panels in the current domain. The user can select one of the displayed panels in order to move to that panel directly without having to navigate intervening panels.

Text Panels

You define a text panel (one that is used to maintain freeform text) within a panel domain by specifying that an element is of type TEXT. No corresponding panel definition is required. CAS provides a text editor facility (that can be invoked through the API) to display and edit this text. See *Text Editor,* on page 2-13 on for details.

See *Chapter 9, Maintaining Panel Domains,* for details of how to define and maintain panel domain definitions.

Help

CAS provides a facility to define and display help text. Help text can be structured in panel format, in simple text format, or using a combination of both. An alias feature is available, which allows you to use the same help text for multiple purposes.

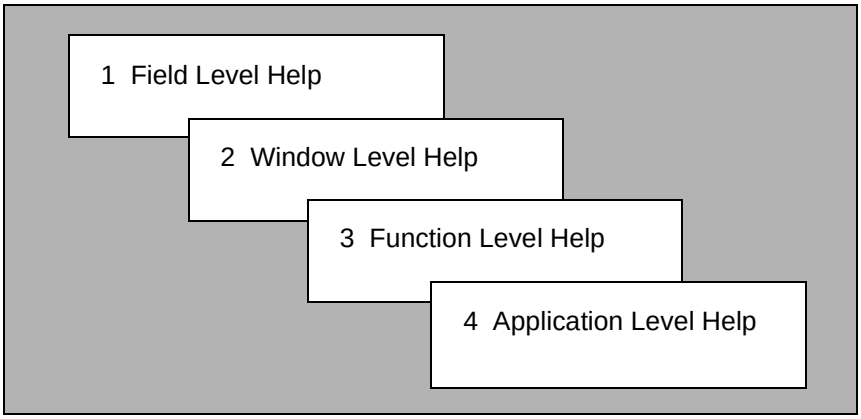
Facilities for constructing help menus, selection lists of help topics, help indexes and tutorials are available. Help text can be merged or copied, both during maintenance and while being displayed.

Help text is associated with a particular application, function, window (a logical area that facilitates context sensitivity) or field.

The Help Hierarchy

There are four levels of help, as shown in Figure 2-3. Help text can be defined at any or all of these levels. The order in which help is presented to the user is levels 1 to 4, that is, beginning with the most specific help available, and becoming more generalized.

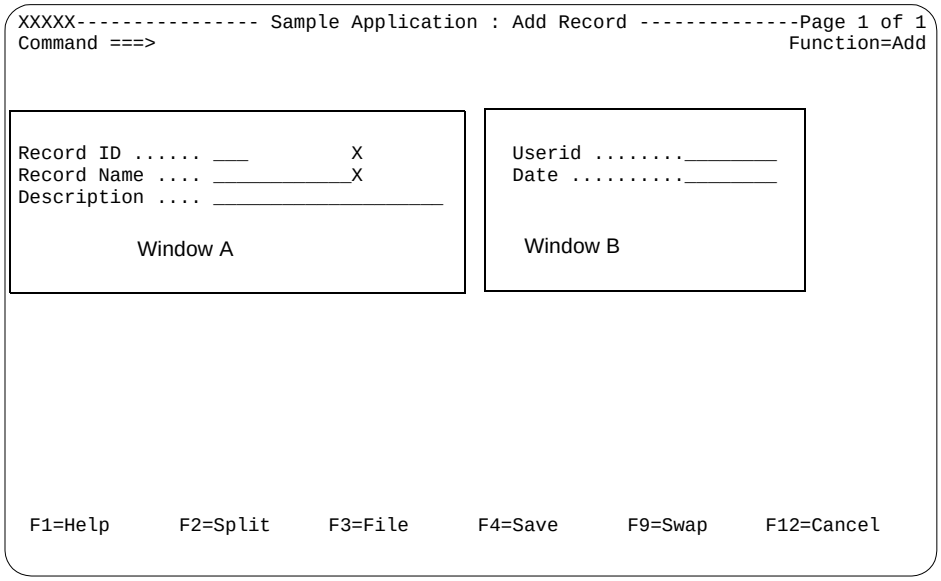
Figure 2-3. The Help Hierarchy



1. **Field Level Help** describes a single field, the data that can be entered by the user, and what happens as a result.
2. **Window Level Help** provides help for a group of logically related fields within a panel. A window is an area that covers part or all of a physical panel. There can be many windows defined for one panel; windows can overlap.
3. **Function Level Help** describes the function that the user is currently performing. Function-level help relates to one particular function within the application (for example, the Add Record function within a maintenance application).
4. **Application Level Help** provides a general overview of the application.

For example, a panel might have two help windows defined as shown in the following diagram:

Figure 2-4. Sample Panel Showing Help Windows



- If a user enters the HELP command from this panel, with the cursor positioned in the **Record ID** field, then CAS displays the field-level help describing the **Record ID**
- If the user then re-enters the HELP command, CAS displays the window-level help for Window A (this file describes the **Record ID**, **Record Name** and **Description** fields)
- Entering the HELP command again displays the function-level help for the Add Record function
- Yet another HELP command displays the application-level help for this application

For details of how to maintain help text, see Chapter 10, *Maintaining Help.*

Messages

A message is a text string which is used to communicate information to the user—for example, an error message. CAS provides facilities for maintaining, building and displaying messages.

Each message has a unique identifier and associated text containing an explanation of the message, the system action and the action that should be taken by the user when the message is displayed.

Variables can be included within messages. When a message is built the variables are substituted into the message to provide specific information relating to the message—for example, an error code.

CAS provides centralized control of messages. All applications call CAS to build messages for display.

CAS message maintenance facilities are described in Chapter 11, *Maintaining Messages.*

Validation

CAS provides facilities to validate data entered in input fields (or any other data) against either:

- A table of defined values
- Predefined rules (for example, a range test or a date format)

Tables

Tables contain a set of entries against which data can be validated.

A table comprises a *table definition* and a number of *table entries* which represent the valid values. CAS provides a facility for creating and maintaining table definitions and entries. See Chapter 12, *Maintaining Tables* for details.

Each table entry can have an abbreviated value, description, and up to ten associated data fields.

Valid values in a table can be specified explicitly as table entries or can be drawn from the following sources:

- A list of attributes defined in an Object Services class
- A list of values in a field of an INFO/MASTER category
- A list of field names in an INFO/MASTER category
- A list of entries supplied dynamically through the CAS API

You can define an exit procedure for a table which performs installation specific processing during table entry maintenance—for example, to validate table entries or restrict the deletion of entries.

Prompted Fields

CAS supports field prompting: if the data to be validated contains a question mark (?), then CAS displays a list of all valid values for the field, with a description of each. The data can contain just part of the value, followed by a question mark; this displays all valid values that generically match the supplied value.

When you use the CAS API to validate data against *a table* this facility is provided by CAS with no additional code required on your part. This facility can be turned off through a parameter to the API (for instance, when you want a question mark to be valid input).

Other Types of Validation

CAS provides other types of validation including alphanumeric, hexadecimal, NCL keyword, and time. You pass the input you want validated to the CAS API and specify the type of validation. You do this by specifying one or more edit numbers in a parameter to the CAS API.

Criteria

A set of criteria is a set of rules that can be used to test a condition. For instance, a set of criteria can be used to select items to go in a list, or items to go on a report, or to validate users' input.

Criteria can simply compare static values or can be complex, combining numerous operators and values. Values can be variable (for example, the current date can be used as a value).

When you recall a set of criteria through the CAS API to perform a test, variable values are supplied by the calling procedure, interactively by the user (through a run-time panel), or by an exit procedure that you define. The criteria exit can process the entries made on a run time panel and also determine whether the panel was actioned or cancelled.

You can specify a data source for the set of criteria, which the exit uses to determine how the criteria are processed. You can define exit parameters within the criteria definition for the exit. This allows the criteria behavior to be changed without having to rewrite the exit procedure and also allows a generalized exit to be written that can then be used by other criteria definitions.

For details of how to define a criteria exit see Chapter 22, *Criteria Exit Procedure Interface.*

MODS provides facilities for maintaining criteria definitions; see Chapter 13, *Maintaining Criteria.*

Commands

CAS provides facilities for defining and processing commands issued by users and by applications.

A command definition contains the unique identifier of the command and an action to be performed when the command is executed.

See Chapter 14, *Maintaining Commands* for details of how to define commands.

The CAS API can be invoked to execute defined commands.

Text Editor

The CAS text editor is a full screen editor that can be included in any NCL application, providing the user with comprehensive editing facilities for up to 9999 lines of text (each of up to 256 characters in length).

- Standard text manipulation facilities are provided via line commands: insert, delete, move, copy, repeat, queue, text split, text flow, and text entry. These are supported singly and as block commands.
- Scrolling functions are supported (FORWARD, BACKWARD, LEFT, and RIGHT), including the ability to specify scroll amounts.
- The ability to find occurrences of a text string is supported, as well as the ability to change one or all occurrences of a nominated text string.
- The ability to position the text on a given line number is also supported.

For detailed information on the editor commands, see Appendix B, *Text Editor Commands*.

You can provide text editing facilities from your application by invoking the text editor through the CAS API and passing the text to be edited. The edited text is returned to the calling procedure.

Text Browse Facility

Text can be displayed in Browse mode, allowing users to view but not update text. This facility can be included in any application, providing comprehensive text browsing facilities for up to 9999 lines of text.

The text browse facility is invoked through the CAS API.

Function Key Areas

The *function key area* (FKA) refers to the bottom two lines of a panel, where function keys and their labels are displayed.

You can modify the labels and actions associated with function keys through the CAS API.

CAS provides predefined function key sets for specific purposes—such as Browse and Update.

The FKA lines are formatted and returned to the calling procedure either when the CAS API is invoked to set key settings or when the KEYS command is executed by the user.

Report Maintenance

Report Writer is an application that is used to define user reports.

Report Writer has the following features:

- A full screen report design and maintenance facility
- A report generation facility
- A scheduler for automating the production of reports
- A maintenance function for defining applications to Report Writer

A report definition is created using a text editor that allows you to define the layout of a report on the screen. Report definitions are stored on a database and can be recalled at any time to produce the report or modify its format and contents.

Report Writer is designed to operate totally independently of the database in which the report data is contained and can create reports from data stored in any database that is defined to it.

A report definition consists of three components:

- **Description**—defines control information about the report such as its name, description and the application that it belongs to
- **Sort Fields**—define the order in which records are sorted on the report
- **Format items**—are lines of text which are printed on the report—each format can consist of any number of lines, made up of both constant and variable data

There are seven report items which are defined for each report:

- **Report Header**—defines material printed at the top of the report
- **Page Header**—defines material printed at the top of every page
- **Data Formats**—define material printed for each record which is passed to Report Writer—there can be multiple data formats.

  An NCL exit procedure can be used to determine which data format or group of data formats is to be printed for each individual record—if there is no exit procedure all data formats are printed.
- **Control Break Headers**—used to print headings above groups of data—control break headers can be printed each time a field on which the data is sorted changes value

- **Control break Trailers**—used to print trailers below groups of data—control break trailers can be printed each time a field on which the data is sorted changes value and are commonly used for printing sub-totals and totals
- **Page Trailer**—defines material printed at the bottom of every page
- **Report Trailer**—defines material printed at the end of the report

Data is secured against illegal access by using the User Access Maintenance Subsystem (UAMS) facility of the SOLVE management services. Report Writer interfaces to Print Services Manager (PSM) for the management of report output.

Mapping Services

Mapping Services is a SOLVE facility that gives NCL access to complex data structures.

What is Mapping Services?

Mapping Services is designed to separate the application's data processing requirements from a need to understand the actual organization of the data.

It means that NCL procedures deal with the logical relationships and usage of data (the data *protocol*), while the system manages and maintains the physical representation of the data (the data *format*).

The Need for Mapping Services

Conventional NCL processing deals with simple data items accessed as NCL variables (or tokens). If a number of variables are logically related, the programmer must understand, through naming conventions or other disciplines, how data items are related and how they must be managed.

Difficulties arise in a number of circumstances, such as where data to be processed by NCL is sourced externally, or where NCL must define an interface to some other processing system. In such cases data is exchanged across a program interface, according to a strict protocol, and can be conveniently represented as some sort of *protocol data unit.* Such a protocol data unit is usually composed of one or more logical components, but must be presented across the interface as a series of bytes, and hence is structured according to some encoding technique.

Nearly all management data conforms to this simple model; however, there are many different encoding techniques employed. These range from very simple rules involving fixed length fields, one following another, to more complicated rules involving variable length structures, and even more complex rules involving self-defining lengths, tags, or similar structures.

The use of variable length data items, and tagged data structures, is popular because it encourages programming precision, and provides a continuous upward migration path. By extending the length of existing structures, or inventing new ones within a data unit, it can retain its original character while evolving to keep pace with new requirements.

Maps and Mapped Data Objects

Mapping Services provides NCL with a logical view of data, while removing from NCL the requirement to understand the physical representation of data. It does this by interposing a map which is used to interpret the data. A map contains the definition of the logical components of a complex data structure, as well as the data's physical representation.

A complex data structure is processed by NCL as a Mapped Data Object (MDO). The NCL procedure can connect an MDO to a map. Only through the map can the NCL procedure access the logical components of the MDO. Once the map connection is made, the NCL procedure can reference the logical entities contained within the MDO by their symbolic names defined within the map.

Each time NCL references a symbolic name, Mapping Services locates the definition within the map, determines how the component is organized, and with a knowledge of the physical representation, navigates the data structure to access the logical data items.

Using this technique, only the system need be aware, through definitions contained within the map, of the actual representation of the data. The NCL programmer, and the procedure itself, need only understand and reference the logical data components.

By modifying the map alone, it is possible to alter the underlying physical data definitions as managed by the Mapping Services facility, without having to change any NCL code.

The SOLVE Map Library

Maps comprise a number of definition records, and are compiled to a loadable form. Map source can be kept in any convenient source library in a similar fashion to NCL procedures.

All compiled maps are lodged in a file that serves as the Map Library. From this library the loadable form of a map can be accessed on demand. Normally a map is loaded only when specifically requested for use by an NCL procedure. When no longer required, it can be deleted from storage by the system.

The map maintenance facilities are described in Chapter 25, *Maintaining Maps*.

For further information on the use of maps in NCL procedures, see the *Network Control Language User's Guide* and the *Network Control Language Reference*.

MODS Administration Facilities

This section provides an overview of MODS administration facilities. Use these facilities to control the storage and maintenance of:

- Panel definitions in panel libraries and the panel paths to be used to retrieve panel definitions
- MODS component definitions in the MODS Control File and the concatenation path to be used when retrieving definitions

Panel Library Maintenance

Panel definitions are stored in VSAM datasets for fast retrieval and update. A VSAM dataset containing panel definitions is called a *panel library.* SOLVE supports multiple panel libraries.

Individual panel definitions are referred to as *library members*. To maintain panel definitions, see Chapter 6, *Maintaining Panels.*

A library can be used as the sole source of panel definitions, or it can be concatenated with other libraries defined to the system. A concatenation of libraries is called a *panel path*. Each user can be defined to use a different path.

The administration functions allow you to:

- Define and maintain panel libraries
- Copy panels between libraries on different paths

MODS Control File Maintenance

A MODS control file contains the following records:

- Application dependent components:
  - $AR—Application Definitions
  - $HM—Help
  - $MH—Menus
  - $MS—Messages
  - $VM—Tables
  - $RW—Reports
  - $LH—Lists
  - $CR—Criteria

- $PV—Panel Domains
- $OS—Object Specifications
- $LD—Language Services Definitions

- Common (non-application dependent) components:
  - $CM—Commands
  - $PS—Print Services Manager (PSM) definitions
  - $LD—Language definitions

Control File Maintenance Facilities

The control file maintenance facilities allow you to:

- Copy records from one control file to another (for example, from test to production)
- Delete records from a control file
- Compare two control files, creating a difference file
- Apply this difference file to one of the control files, making it identical to the other control file
- Browse the records stored on a control file
- Search records stored on a control file

For details, see Chapter 17, *Maintaining the MODS Control File*.

Control File Concatenation

Control file concatenation allows multiple control files to be accessed concurrently via the concatenation path, and provides a distinct separation between distributed, production, and test definitions in order to simplify maintenance and provide control over the development environment.

Concatenation Path

The concatenation path can contain up to five separate files. The concatenation path to be used by an SOLVE region is defined during initialization in the NMINIT procedure. See the distributed NMINIT procedure for further information.

Merged View

All MODS component lists display a merged view of all files in the path. This means that the distinctions between individual files are ignored and the contents of all files in the concatenation path are eligible for selection.

In the case of duplicate components present at different levels in the path, the topmost instance has precedence over those at lower levels. The identifier of the file in which the component is located is displayed on the right-hand side of component lists.

Automatic Promotion

Only the topmost file in the path is modifiable by means of any of the maintenance facilities. A request to update any definition not resident in the top file retrieves the definition from the lower file and stores it into the topmost file (this is known as automatic promotion). New records are always added to the topmost file in the path.

Manual Demotion (Lodgement)

To move any definition downward in the path, the MODS : Definition Utilities must be used and the source and destination files specifically named.

These utilities also provide a means of viewing the contents of files in isolation. That is, they do not present a merged view, but list only those components that are actually contained in the named file.

Two major benefits flow from control file concatenation:

- Operational Flexibility

  The ability to give multiple groups of users of the same application different views of that application. A *view* in this context constitutes both presentation and behavioral differences. Views can be used to implement significant usage differences, such as TEST and PRODUCTION.

- Maintenance Advantages

  Library concatenation greatly eases maintenance difficulties and facilitates better change control with inbuilt backout capability. It also allows management of changes to distributed applications.

Sharing Control Files

MODS control files can be shared between regions where multiple SOLVE regions exist (for example, a production and test region). You must ensure, however, that only one of these regions is capable of updating records stored on the shared control files. Data corruption can occur if this is not done.

Update capability is specified when defining the control file path in the NMINIT procedure for each SOLVE region.

SOLVE Web File System

The SOLVE Web file system contains all the files used by the SOLVE Web Interface. The Web file system is stored in the MODS file; however, it has no relation to the other MODS components. For more information on the Web file system, see Appendix F, *SOLVE Web File Utilities*.

3

Getting Around

This chapter describes the primary menus from which you can access MODS facilities.

Accessing MODS Facilities

The MODS menu structure is displayed in Figure 3-1.

All facilities described in this manual are accessed through the MODS : Primary Menu, as illustrated in Figure 3-2. You can reach this menu by selecting option **D** from the SOLVE : Primary Menu.

Figure 3-1. MODS Facilities Menu Structure

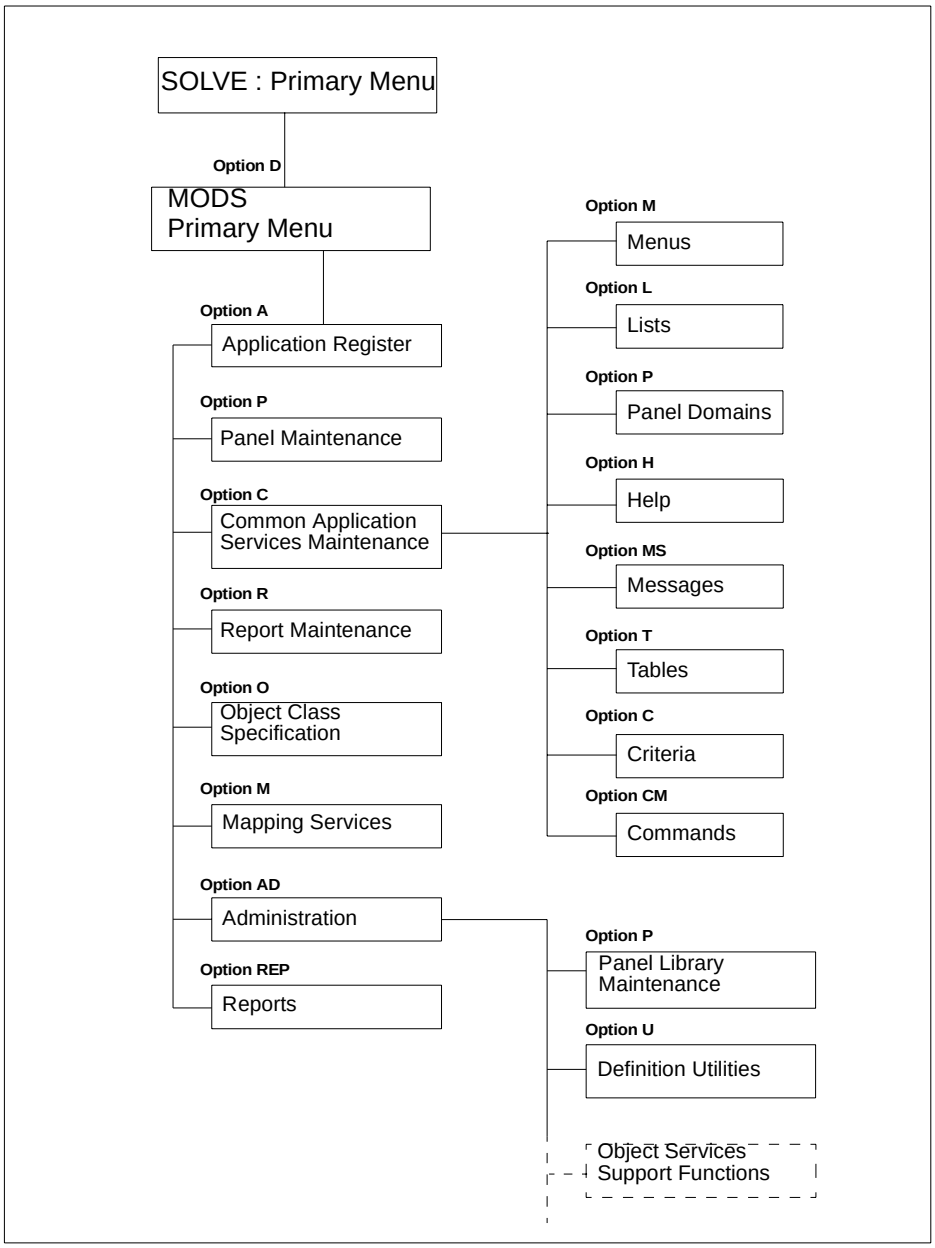
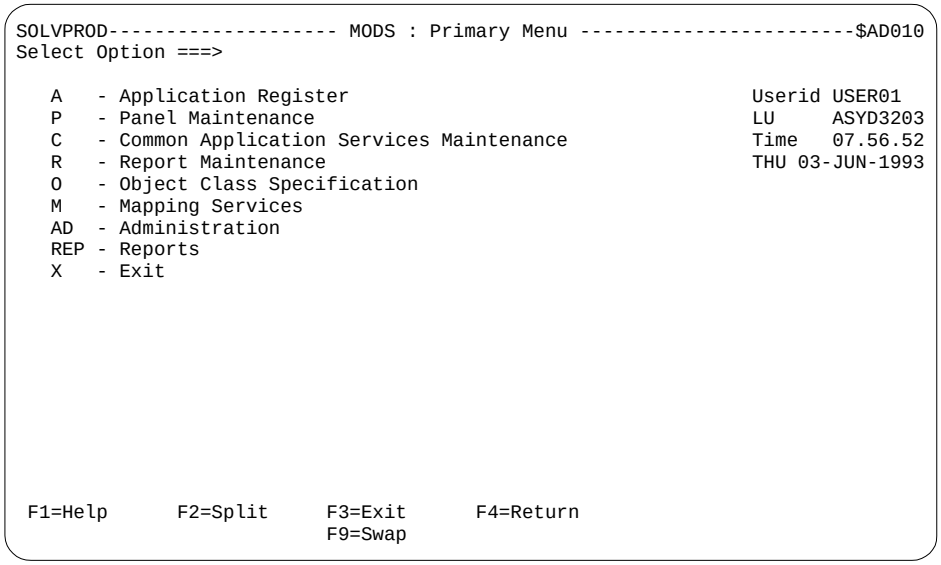


Figure 3-2. MODS : Primary Menu



The options displayed on the MODS : Primary Menu are as follows:

A – Application Register

This option enables you to maintain application definitions, as described in Chapter 4, *Registering an Application.* The first step when creating an application is to define the application in the Application Register.

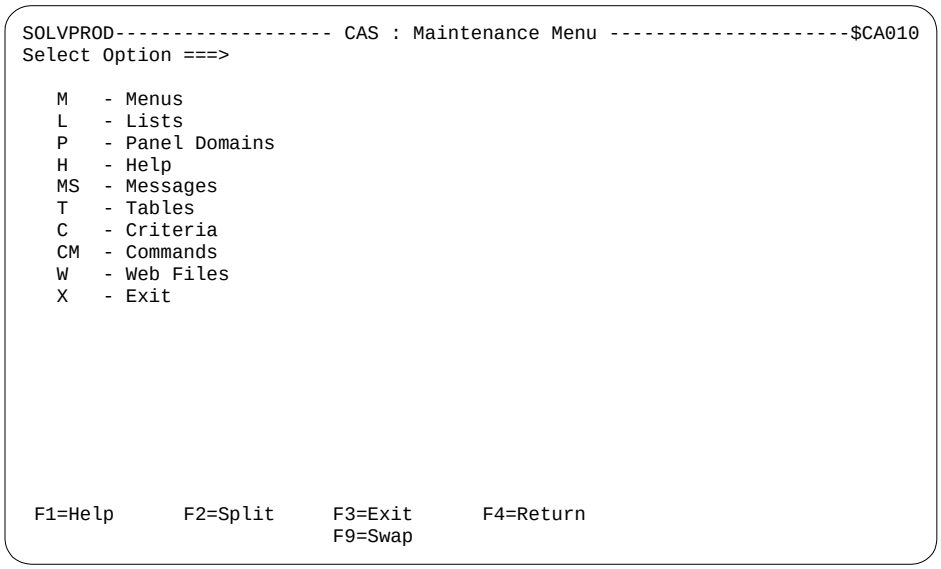
P – Panel Maintenance

Individual panels can be maintained as described in Chapter 6, *Maintaining Panels.*

C – Common Applications Services Maintenance

This option displays the CAS : Maintenance Menu (see Figure 3-3) used to maintain CAS application components. The CAS maintenance facilities are described in chapters 7 to 14.

Figure 3-3. CAS : Maintenance Menu



R – Report Maintenance

This option displays the Report Writer : Report Definition Menu, enabling you to add, update, delete, copy, list and view report definitions. See the *Report Writer User's Guide* for details.

O – Object Class Specification

This option enables you to maintain object class specifications. This facility is available only for customizing SOLVE:Central applications. For details, see the *SOLVE:Central Implementation and Administration Guide* and the *SOLVE:Central Customization Guide.*

M – Mapping Services

This option allows you to maintain ASN1 maps. See Chapter 25, *Maintaining Maps*, for details.

AD – Administration

This option displays the MODS : Administration Menu, giving you access to:

- Panel library maintenance facilities. See Chapter 16, *Maintaining Panel Libraries*, for details.
- MODS control file maintenance facilities. See Chapter 17, *Maintaining the MODS Control File*, for details.
- Object Services administration facilities. These facilities are available only for the SOLVE for systems administration applications. See the *SOLVE:Central Implementation and Administration Guide* and the *SOLVE:Central Customization Guide* for details.

REP – Reports

This option displays a list of predefined MODS reports. Reports can be selected from the list for generation. For further information, see Chapter 15, *Printing MODS Components*.

MODS Access Authority

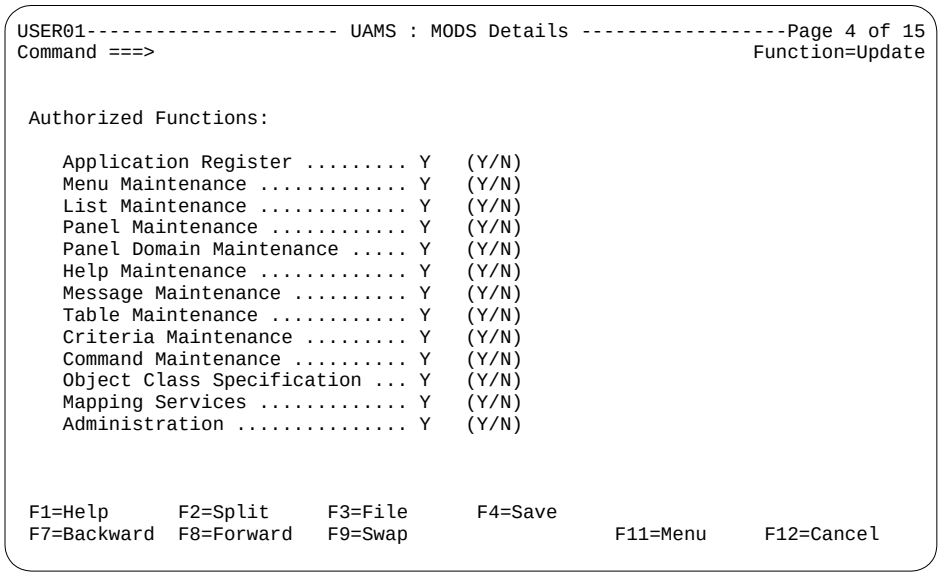
To access the MODS : Primary Menu, you need to have the **Managed Object Dev. Services** field on the UAMS : Access Authorities panel set to Y, as shown in Figure 3-4.

Figure 3-4. UAMS : Access Authorities Panel

```
USER01------------------- UAMS : Access Authorities ---------------Page 3 of 15
Command ===>                                                    Function=Update


 Authorized Functions:

    Network Management ........... Y   (Y/N)
    Systems Management ........... Y   (Y/N)
    Information Management ....... Y   (Y/N)
      INFO/MASTER Maintenance .... N   (Y/N)
    Automation Services .......... Y   (Y/N)
    Access Services .............. Y   (Y/N)
    Data Transfer Services ....... Y   (Y/N)
    Electronic Mail .............. Y   (Y/N)
    Operator Console Services .... Y   (Y/N)
    Management Services .......... Y   (Y/N)
       UAMS Maintenance .......... Y   (Y/N)
       Broadcast Services ........ Y   (Y/N)
       Object Services Support ... Y   (Y/N)
          Object Services Security N   (Y/N)
       System Support Services ... Y   (Y/N)
    Managed Object Dev. Services . Y   (Y/N)
 F1=Help      F2=Split     F3=File      F4=Save
 F7=Backward  F8=Forward   F9=Swap                  F11=Menu     F12=Cancel
```

If the **Managed Object Dev. Services** field is set to **YES**, then the UAMS : MODS Details panel is the next panel displayed (shown in Figure 3-4).

Figure 3-5. UAMS : MODS Details Panel



Each of the MODS facilities is displayed on this panel. You can specify that a user has access to a given facility by entering a **Y** in the field next to the facility.

All authorizations shown on the UAMS : MODS Details panel default to **N**.

Part II

Maintaining Application Components

4

Registering an Application

An application is a group of logically related functions and/or data. Before an application's components can be defined, an application definition must be created in the application register.

This chapter describes the facilities available for adding and maintaining application definitions.

About Application Definitions

An application definition consists of a three-character identifier, a descriptive name, a one- to eight-character message prefix, and, optionally, some comments.

The application identifier (or application ID) provides a unique naming space for an application's components—it prefixes the identifiers of all these components. This makes it easy to find all components that are used by a particular application and simplifies maintenance of component definitions.

See Chapter 5, *Naming Standards*, for details regarding the selection of an application ID and a message prefix.

Defining an Application Definition

You can create an application definition by adding a new definition (either by using the appropriate menu option or by using the ADD key on a list) or by copying an existing definition.

The Application Definition Menu

Application definitions are accessed through the MODS : Application Definition Menu (option **A** on the MODS : Primary Menu).

This menu allows you to add, browse, update, delete, or copy application definitions, or to obtain a list of application definitions.

The MODS : Application Definition Menu is illustrated in Figure 4-1.

Figure 4-1. MODS : Application Definition Menu

```
SOLVPROD------------- MODS : Application Definition Menu ----------------$AR010
Select Option ===>

   A   - Add Application
   B   - Browse Application
   U   - Update Application
   D   - Delete Application
   C   - Copy Application
   L   - List Applications
   X   - Exit

Appl ID ... ___ ( Required B U D C  Optional A L )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The field displayed on the MODS : Application Definition Menu is as follows:

Appl ID

This three-character identifier is used to uniquely identify the application, and to tag all components that belong to it. You must make an entry in this field when using the Browse, Update, Delete, and Copy options, to specify the application definition to which the function is applied.

If you select the list option and make an entry in this field, the list is restricted to those application definitions that generically match the entry you make.

The following options are available on the Application Definition Menu:

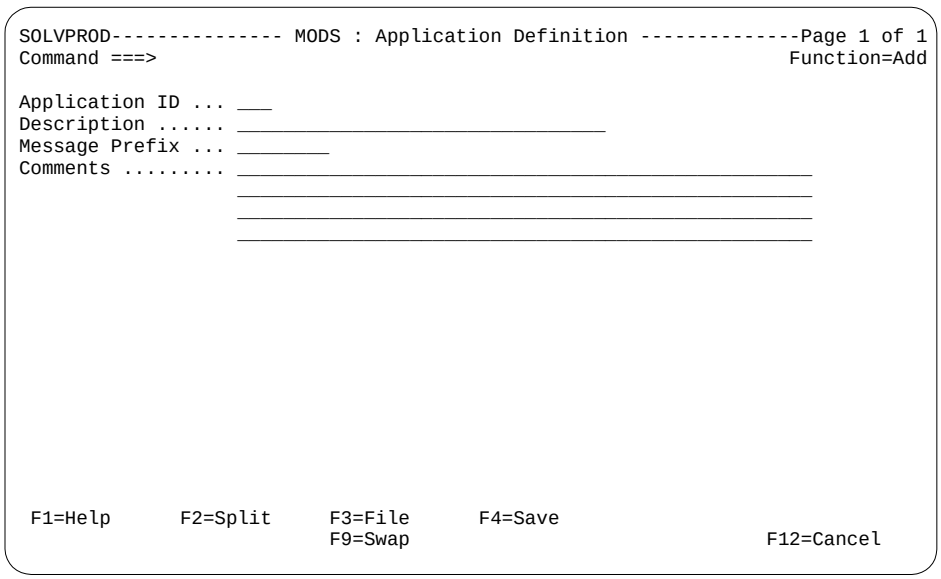
Option	Explanation
A	Add a new application definition
B	Browse an application definition
U	Update an application definition
D	Delete an application definition
C	Copy an application definition
L	List application definitions

A detailed description of these options follows.

Adding an Application Definition

Option **A** displays the MODS : Application Definition panel, as shown in Figure 4-2.

Figure 4-2. MODS : Application Definition Panel



The fields displayed on the MODS : Application Definition panel are as follows:

Application ID

This three-character identifier is used to uniquely identify the application, and to tag all components that belong to it.

Description

A 1- to 32-character description of the application.

Message Prefix

A 1- to 8-character identifier that prefixes all messages belonging to the application.

Comments

A more detailed description of the application. Although this is an optional field, it is recommended that comments be included to assist the administrator when maintaining application definitions.

When you have completed the application definition, press the FILE key. To cancel the definition, press the CANCEL key.

Maintaining Application Definitions

Use the options described in this section to maintain existing application definitions. You can select these options from the MODS : Application Definition Menu or from a list of application definitions.

Browsing an Application Definition

Select option **B** to browse an application definition. The MODS : Application Definition panel is displayed, as shown in Figure 4-2, except that in this case the function is Browse and the fields cannot be modified.

Updating an Application Definition

Select option **U** to update an application definition. The MODS : Application Definition panel is displayed, as shown in Figure 4-2, except that in this case the function is Update and the **Application ID** field cannot be modified. Update this as required; see the section, *Adding an Application Definition*, on page 4-3, for details.

When you have finished updating the application definition, press the FILE key. To cancel the update, press the CANCEL key.

Deleting an Application Definition

Select option **D** to display a message requesting you to confirm the deletion. Press ENTER to delete the application definition, or press the CANCEL key to cancel the deletion.

Copying an Application Definition

To copy an existing application definition to another (new) application definition, select option **C** from the MODS : Application Definition Menu, and provide the **Appl ID** field of the application definition to copy from (or, alternatively, select the copy action against an application definition in a list).

The MODS : Application Definition panel, as shown in Figure 4-2, is displayed, with the fields containing the values of the copied definition. Modify the new application definition as required (see the section, *Adding an Application Definition*, on page 4-3, for details).

Listing Application Definitions

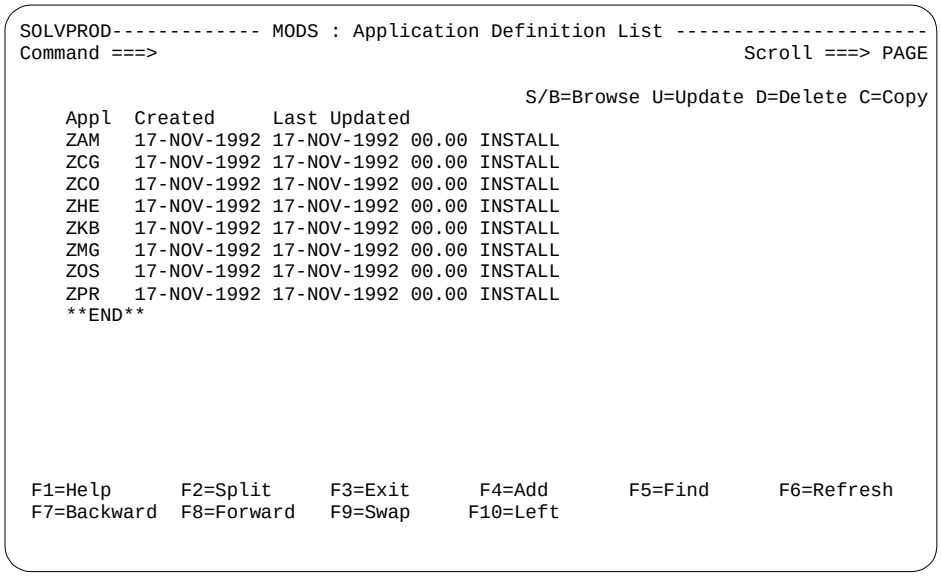
Select option **L** to list application definitions. If you make an entry (or partial entry) in the **Appl ID** field when you select this option, then the list of application definitions is restricted to those definitions that match the entry you make.

The MODS : Application Definition List is displayed over two panels, as shown in Figure 4-3 and Figure 4-4. Use the RIGHT and LEFT keys to scroll between the panels of the list.

Figure 4-3. MODS : Application Definition List (page 1)

```
SOLVPROD------------ MODS : Application Definition List ----------------------
Command ===>                                                 Scroll ===> PAGE

                                        S/B=Browse U=Update D=Delete C=Copy
    Appl  Description                   Msg Prefix                File ID
    ZAM   SOLVE:Asset                   ZAM                       MODSDIS
    ZCG   SOLVE:Change                  ZCG                       MODSDIS
    ZCO   SOLVE:Configuration           ZCO                       MODSDIS
    ZHE   SOLVE:Problem Help Desk       ZHE                       MODSDIS
    ZKB   SOLVE:Problem Knowledge Base  ZKB                       MODSDIS
    ZMG   SOLVE for Systems Administration ZMG                    MODSDIS
    ZOS   Object Services               ZOS                       MODSDIS
    ZPR   SOLVE:Problem                 ZPR                       MODSDIS
    **END**









 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 4-4. MODS : Application Definition List (page 2)



The fields displayed on the MODS : Application Definition List are as follows:

Appl
The identifier of the application definition.

Description
A brief description of the application.

Msg Prefix
The prefix that is used for messages defined for this application.

File ID
The identifier of the MODS control file in which the application definition is held.

Created
The date the application definition was created.

Last Updated
The date and time that the application definition was last updated and the user ID of the person who performed the update. If the application definition has not been modified since installation, then INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B, S, or /	Browse the selected application definition
U	Update the selected application definition
D	Delete the selected application definition
C	Copy the selected application definition

These actions can also be performed by choosing the corresponding option from the Application Definition Menu.

Maintaining Application Groups

Any application that uses lists and reports must have its own copy of the table $ADGROUP in which the names of associated groups are defined. The $ADGROUP table is currently available only to CAS List Services and Report Writer. Regardless of which application contains the table, $ADGROUP's attributes must always be defined as shown in the example in Figure 4-5 (except for the **Appl ID** and **Field description** fields, which need to be changed for the relevant application).

Figure 4-5. Sample Group Table Definition

```
SOLVPROD------------------ CAS : Table Description -----------------Page 1 of 2
Command ===>                                                  Function=Browse

Appl ID .................. $AD
Field name ............... $ADGROUP
Field description ........ Sample Group Defn.
Edit type ................ TABLE (TABLE, OSATT, OSFLD, IMFLD or IMREC)
For Edit type = TABLE:
   Validation exit ................. $CAVM040
   Sequence numbers ................ NO  (YES or NO)
   Load table? ..................... YES (YES or NO)
   Max abbreviation length .........     (3 - 8 or blank if none)
   Max full value length ........... 12  (3 - 20)
   Max description length .......... 38  (3 - 38 or blank if none)
For Edit type = IMFLD or IMREC:
   INFO/MASTER category ............
For Edit type = IMREC:
   INFO/MASTER field ..............
   INFO/MASTER description field ...
For Edit type = OSATT or OSFLD:
   Object Services Class ID ........

 F1=Help      F2=Split     F3=Exit                              F6=Entries
              F8=Forward   F9=Swap
```

See Chapter 12, *Maintaining Tables*, for details of how to maintain table entries.

5

Naming Standards

This chapter describes naming standards that must be used for applications that you develop.

Note

These standards do not generally apply when you are customising a supplied product. Where customization is permitted for a given product the specific standards that apply are detailed in the relevant product documentation.

Naming Standards

The primary method of applying naming standards is through an application identifier.

An application is a group of logically related functions and/or data. Every application is defined in the Application Register and named with a unique identifier. This *application identifier* provides a unique name space for application components.

Before you can define application components, you must define an application definition within the Application Register.

The following sections detail how individual application components are named.

Application Definitions

The first character of the application identifier for applications that you define must be Y.

The message prefix, specified within an application definition, must be the same as the application identifier.

Panels

The first three characters of the Panel identifier must be the same as the application identifier.

NCL Procedures

The first three characters of the procedure name must be the same as the application identifier.

CAS Components

The application identifier must be explicitly specified for Menus, Lists, Panel Domains, Help, Tables, and Criteria Definitions.

Note

The help function names OVERVIEW and INDEX are reserved for application overview help and the help index, respectively.

The first three characters of Message and Command identifiers must be the application identifier.

Object Class Specifications

Object Services facilities are not available for customer applications.

Reports

The first three characters of a report application identifier must be the same as the application identifier.

The Report Application Identifier must be explicitly specified within Report Definitions.

Map Definitions

The first three characters of a Map Name must be the application identifier.

Externally Visible Entities

In addition to application components there are other entities that are visible outside an application and therefore require name space protection by prefixing their names with the application identifier. These include the following:

- Global variables
- Global vartables
- Locks
- Variables/MDOs that are shared between applications
- Data Domain names
- File identifiers
- NDB identifiers
- EDS events
- NDB Global Format names

6

Maintaining Panels

This chapter describes the MODS : Panel Maintenance facilities for creating, maintaining, and customizing full-screen panel definitions. The section titled *Defining and Maintaining Panels* describes these facilities, while the section titled *Designing Panels* describes the actual content of a panel definition.

About Panel Maintenance

Panel definitions can be used by NCL processes executing in any NCL environment associated with a display window. The panels enable NCL processes to display output data, and can be designed with input fields for communicating back to these NCL processes.

For details of how NCL processes use panel definitions, see the chapter, *Designing Interactive Screens (Panel Services)*, in the *Network Control Language User's Guide*.

Panels can be added to a panel domain to control how a group of panels is displayed during data entry. For details on how to associate panels with a panel domain, see Chapter 9, *Maintaining Panel Domains*.

Panels are created and changed using an online editor. See Appendix C, *Panel Editor*, for detailed information on using the editor. You must be authorized to use this facility, and installations can limit the number of concurrent edit users by restricting the amount of storage available to the editor.

The SOLVE management services standard split-screen facilities are very useful when designing panels: the panel editor can be used on one window, while the panel view facility of MODS : Panel Maintenance displays the current version of the panel being developed on the other.

Concepts and Terminology

Panels are stored in VSAM datasets for fast retrieval and update. A VSAM dataset containing panel definitions is called a *panel library*, with individual panel definitions being termed *members* of the library. Each member maintains details on the date of creation, the date of last modification, the user ID, the number of statements in the member, and the modification level.

The SOLVE management services support multiple panel libraries. A library can be used as the sole source of panel definitions, or it can be concatenated with other libraries defined to the system. A concatenation of libraries is called a *panel path*. When a user is defined, the panel path they use is defined. Each user can be defined to use a different path. The default path is called PANELS. Panel paths are further described in the chapter titled *Access and Security* in the *SOLVE Management Services Implementation and Administration Guide*.

When a library is defined, the person defining it can determine if the library can be edited on the system. When a path is defined, the definer can choose to allow or disallow edit of selected libraries in the path.

Note

To update panel definitions in a library using the MODS panel maintenance facilities, the library must have edit allowed by both the library *and* path definitions.

For details of allocating, defining, and removing panel libraries, see Chapter 16, *Maintaining Panel Libraries*.

Retrieving Panels From Panel Libraries

When required, panel specifications are retrieved from a library in the user's current path.

To eliminate overheads associated with retrieving the panel from the library, an in-storage queue of active panels is maintained. When a panel is first referenced it is retrieved from a panel library and stored on the active panel queue.

Thereafter, the panel is retrieved from the active panel queue without reference to the panel's library. If one of these panels is modified (using the online editor), any old copy is removed from the active panel queue so that the next reference retrieves the updated panel.

Note

If a panel library is being shared by more than one SOLVE system, a modified panel is only removed from the active panel queue of the SOLVE management services on which the panel change has been made. The other SOLVE systems continue to use the old panel until it is rolled off the active panel queue by other panels being used in the system. The LIBRARY REFRESH command can be used to drop all panels loaded from a library from the active panel queue.

The number of panels that can be retained on the active panel queue can be tailored by the installation using the SYSPARMS command MAXPANEL operand. This is described in the *SOLVE Management Services Implementation and Administration Guide*, in the chapter titled *Customizing SOLVE Management Services*.

Defining and Maintaining Panels

You can create a new panel definition by selecting the Add Panel option from the MODS : Panel Maintenance Menu, or copy an existing panel definition by using the Copy panel option. See *Copying a Panel Definition*, on page 6-11. You can browse, update, delete, list, view, print, rename, display, and move an existing panel.

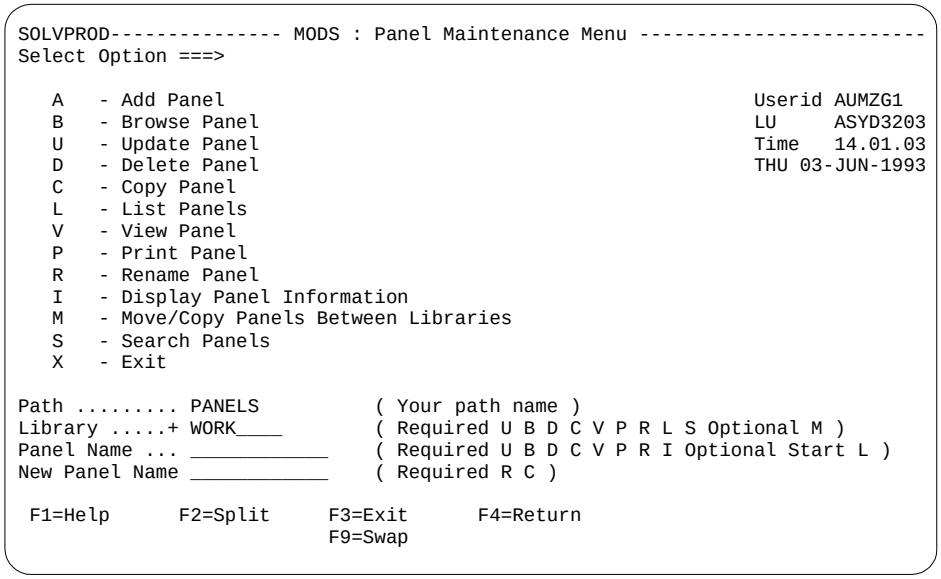
The MODS : Panel Maintenance Menu

The MODS : Panel Maintenance Menu can be accessed by selecting option **P** from the MODS : Primary Menu.

Alternatively, you can invoke the $EDMENU NCL procedure from an OCS screen or a similar command screen. (The original OCS window is restored when you exit from the editor session.)

The MODS : Panel Maintenance Menu is illustrated in Figure 6-1.

Figure 6-1. MODS : Panel Maintenance Menu



All panel maintenance facilities operate within the *path* defined for your user ID. The editing options shown on the MODS : Panel Maintenance Menu allow you to add, update, list, delete, or rename panels, or copy the source data for online panels within libraries or between libraries in your path. You can also display, test, or print existing panels.

Your library path is the name of the panel library path defined for you in your user ID definition. The default path name is PANELS.

The following fields are displayed on the MODS : Panel Maintenance Menu:

Library

The name of the library that contains the panel you want to access. If you are uncertain of the library name, enter a question mark (?) and press ENTER for a list of libraries allowed (see Figure 6-2). You can then select a library from this list for targeting the member you need.

Panel Name

The name of the panel you want to access. The panel name is a unique 1- to 12-character name, which is nominated whenever a request is made to display the panel.

New Panel Name

When you are using the Rename Panel option or the Copy Panel option, this field must contain the new panel name.

If you enter a question mark (?) in the **Library** field on the MODS : Panel Maintenance Menu or the MODS : Panel Move/Copy Menu when you select an option, the MODS : Panel Library List, as shown in Figure 6-2, is displayed.

Figure 6-2. MODS : Panel Library List

```
SOLVPROD----------------- MODS : Panel Library List --------------PATH : PANELS
Select Entry ===>                                                Scroll ===> PAGE

                                                                 Select a Library
         Path: PANELS   Description: Main Panel Path
         Library-Name   Edit       Description
   1    - WORK           YES        ONGOING DEVELOPMEMT
   2    - USRPANL        YES        USER PANELS
   3    - TEST           YES        DEVELOPMENT LODGED TO TEST
   4    - DISPANL        NO-LIB     V3.0 DISTRIBUTED PANELS
          **END**









 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap
```

All libraries in the current path are displayed. If you select a library by entering its number in the **Select Entry** field, processing continues as if you had typed the library name yourself on the original menu and pressed ENTER.

The top right corner of the panel shows the path name. The fifth line of the panel displays the path name and its description.

The following information is displayed about each library in the path:

Edit

Tells you if the library can be edited. This field can contain the following:

- **YES** means the library can be edited (unless edit is restricted on this system)
- **NO-LIB** means the library definition prevents this library from being edited
- **NO-PATH** means the path definition does not allow edit of this library

Description

Displays the description of this library.

Select one of the functions displayed on the MODS : Panel Maintenance Menu by entering the letter next to the desired option in the **Select Option** field, and pressing the ENTER key.

Some options require that a library name is specified in the **Library** field. This defaults to the first editable library in your path. You can change this to any other library in your path. If you are unsure what libraries are in your path, enter a question mark (?) next to this field—this displays the MODS : Panel Library List (Figure 6-2 on page 6-5), which is a selection list of all libraries in the current path.

Note

You can only use panel maintenance facilities on libraries in your path.

The following options are available from the MODS : Panel Maintenance Menu:

Option	Explanation
A	Add a new panel definition
B	Browse an existing panel definition
U	Update an existing panel definition
D	Delete an existing panel definition
C	Copy a panel definition
L	List panel definitions
V	View a panel as displayed to the user
P	Print a panel definition
R	Rename a panel definition
I	Display panel information
M	Move/Copy panels between libraries
S	Search panel definitions for a character string

Each of these menu options is described in detail below:

Adding a Panel Definition

Select option **A** to create a new panel definition. Enter the name of the library where you want the panel to be stored in the **Library** field and the name of the panel in the **Panel Name** field on the MODS : Panel Maintenance Menu. The **Library** name entered must be editable on your library path.

This option invokes the editor. The editor screen, as shown in Figure C-1, is displayed ready for the entry of new data. For details of the editor commands, see Appendix C, *Panel Editor*.

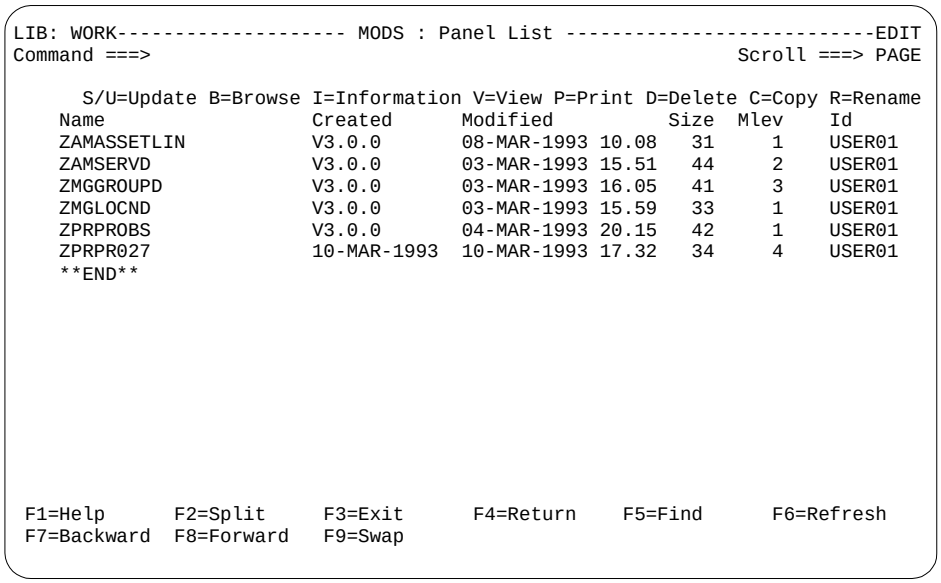
Listing Panel Definitions

Select option **L** from the MODS : Panel Maintenance Menu to display a list of panels defined in a particular library.

Enter the name of the library in the **Library** field on the MODS : Panel Maintenance Menu. The MODS : Panel List panel is displayed (as in Figure 6-3), showing information about each panel defined in the library.

If you make an entry in the **Panel Name** field on the MODS : Panel Maintenance Menu, MODS : Panel List displays the list of panel definitions starting with the panel name that matches the value you enter. (If this is a partial panel name match, the first panel name that partially matches the value that you entered appears as the second item in the displayed list.)

Figure 6-3. MODS : Panel List



The fields displayed on the MODS : Panel List are as follows:

Name

The name of the panel stored in the library. This is the name used by the &PANEL statement which displays the panel.

Created

The creation date of the panel (usually in the format *dd-mon-year*—for example, 26-JAN-1993). In the case of distributed panels, this column can contain the distributed version number, for example V3.0.0.

Modified

The date and time that the panel was last modified. The date is in the format *dd-mon-year* and the time is in hours and minutes. If the panel has never been modified, this column is blank.

Size

The number of lines in the panel definition.

Mlev

The modification level of the panel definition (that is, the number of times the panel has been modified). If the panel has never been modified, this number is zero. The modification level is set to 0 when a member is first created, and then incremented by one each time a new edit session is started for the member. The modification level can be incremented up to 99, after which incrementing stops.

Id

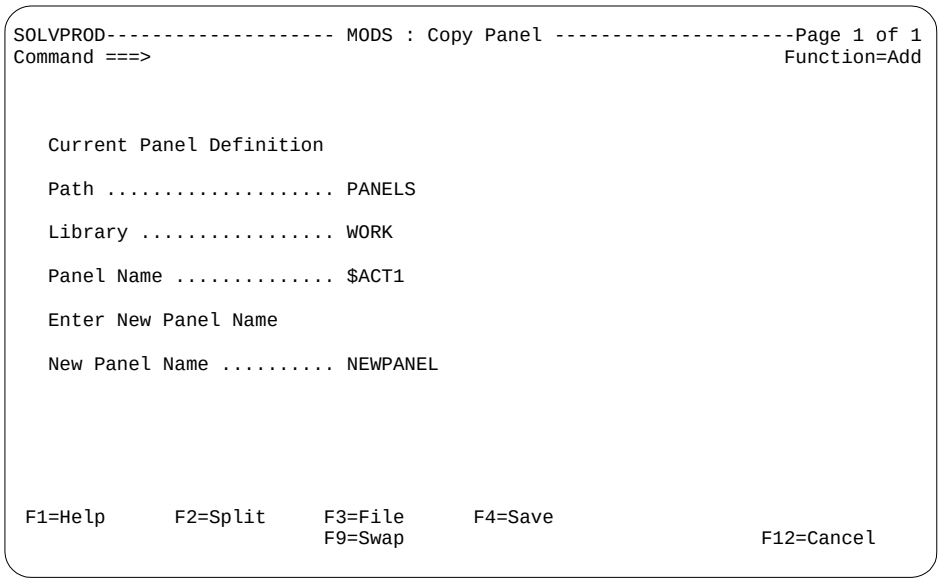
The user ID of the person who last updated (or created) the panel definition. In the case of a distributed panel this column can contain INSTALL as the user ID.

From the MODS : Panel List you can perform the following actions on list members:

Mnemonic	Action
U	Invoke the full screen editor for this member in the current library. For details of the editor commands, see Appendix C, *Panel Editor*.
B	Browse the panel definition.
S	Selects the panel for updating (same as U) if the library is editable; selects the panel for browsing (same as B) if the library is not editable.
I	Obtain Information about the panel. A list of all libraries in the path is displayed in concatenation order. If the member is present in a library, the normal member statistics for the member are displayed next to it. The line for the current library is highlighted. This option is useful for finding out which library a member resides in, and which is the highest in the concatenation order of the path.
V	View a panel definition in the library being listed as it is displayed to the user. This can be useful when developing a panel, to check its appearance. **Note:** Because of the concatenation of panel libraries, this definition might not be the definition used when an NCL procedure issues the &PANEL verb.
P	Print the panel definition. The print data is queued to PSM.
D	Delete the panel definition from this library. You are asked to confirm the deletion: press ENTER to delete the panel definition from this library, or the CANCEL key (F12) to cancel the deletion.

Mnemonic	Action
C	Copy the selected panel to create a new panel. The Copy Panel panel is presented, as shown in Figure 6-4. Enter the new panel name in the **New Panel Name** field. Press the FILE key to proceed with the copy operation. On the Copy panel you can use the SAVE key (rather than file) so that you remain on the panel and can make multiple copies. This action copies the panel within the current library. To copy panels between libraries in the path, use option **M** on MODS : Panel Maintenance Menu. **Note:** To copy a panel you must have edit allowed by both library and path definitions.
R	Rename an existing panel (enter R next to the list member you want to rename). The Rename Panel panel is presented. All of the fields on the Rename Panel panel are the same as those on the Copy Panel panel, which is shown in Figure 6-4. Enter the new panel name in the **New Panel Name** field. Press the FILE key to proceed with the rename operation. **Note:** To rename a panel you must have edit allowed by both library and path definitions.

Figure 6-4. MODS : Copy Panel



Note

If the library is not editable, then the Update, Delete, Copy, and Rename options are not available. A library might not be editable either because edit of this library is disallowed, or because editing was restricted from the current system by the SYSPARMS EDITMAXK=NONE command.

These actions correspond to the menu options described earlier in this chapter. Some actions, however, operate differently to the menu option—these are described below.

Special List Commands

In addition to the standard commands available with all lists, the following special commands are available on the MODS : Panel List (Figure 6-3):

S *panelname*

This command allows you to select a new or existing panel for update, without having to scroll to the correct place in the selection list. Similarly, any of the other line selections (except Delete) can be entered as primary commands if followed by the panel name. For example, you can view a panel named MYPANEL by entering **V MYPANEL** on the command line.

SORT

This primary command can be used to change the sort order of the list. The list is normally sorted by name, but the sort command can be used to sort it by any of the other columns displayed on the list. For example, **SORT CRE** can be used to sort the list by creation date.

Secondary sort fields can also be specified. For example, **SORT ID SIZE** sorts the list by the name of the user who last updated the panel, and by size for each user ID. The valid sort fields are **Name**, **Created** (or **Cre**), **Modified** (or **Mod**), **Mlev**, and **ID**. The sort process can take some time if you are listing a large panels dataset.

Browsing a Panel Definition

Select option **B** to browse a selected panel definition. Enter the name of the library where the panel is stored in the **Library** field and the name of the panel in the **Panel Name** field on the MODS : Panel Maintenance Menu.

If the panel does not exist in the library specified, then an error message is displayed; otherwise the selected panel definition is displayed.

Updating a Panel Definition

Select option **U** to update an existing panel definition. Enter the name of the library where the panel is stored in the **Library** field and the name of the panel in the **Panel Name** field on the MODS : Panel Maintenance Menu. The Library name entered must be editable on your library path.

This option invokes the editor; the edit screen displays the existing panel data. For details of the editor commands, see Appendix C, *Panel Editor*.

Deleting a Panel Definition

Select option **D** to delete a panel defined in a particular library. Enter the name of the library in the **Library** field and the name of the panel you want to delete in the **Panel Name** field on the MODS : Panel Maintenance Menu. The library must be editable on your path.

A panel is displayed asking you to confirm the delete operation. Press ENTER to delete the panel definition, or press the CANCEL key to stop the deletion.

Copying a Panel Definition

Select option **C** to copy a panel within a library. You must enter the name of the new panel in full in the **New Panel Name** field on the MODS : Panel Maintenance Menu.

If the name already exists, an error message is displayed and the copy function aborted. If the copy is successful, the message `PANEL COPIED SUCCESSFULLY` is displayed.

Viewing a Panel Definition in Display Format

Use option **V** to see what a panel looks like when displayed by an NCL procedure. On the MODS : Panel Maintenance Menu, enter the name of the library where the panel is stored into the **Library** field, and the name of the panel in the **Panel Name** field.

The view function displays the panel in the nominated library. Note that the displayed panel is not necessarily the panel which would be displayed by any NCL user with the same panel library path. A panel with the same name can be in another library in the path.

Printing a Panel Definition

Select option **P** from the MODS : Panel Maintenance Menu to print a specific panel definition. Enter the name of the library where the panel is stored into the **Library** field and the name of the panel in the **Panel Name** field on the MODS : Panel Maintenance Menu.

If the panel does not exist in the library specified, an error message is displayed. Otherwise the panel definition is printed using Print Services Manager (PSM).

See the chapter titled *Print Services Manager (PSM)* in the *SOLVE Management Services User's Guide* for further details.

Renaming a Panel Definition

To rename a panel within a library, select option **R** from the MODS : Panel Maintenance Menu. The new name must be entered in full in the **New Panel Name** field on the MODS : Panel Maintenance Menu.

If the name already exists, an error message is displayed and the rename function aborted. If the rename is successful, the following message is displayed:

```
PANEL RENAMED SUCCESSFULLY
```

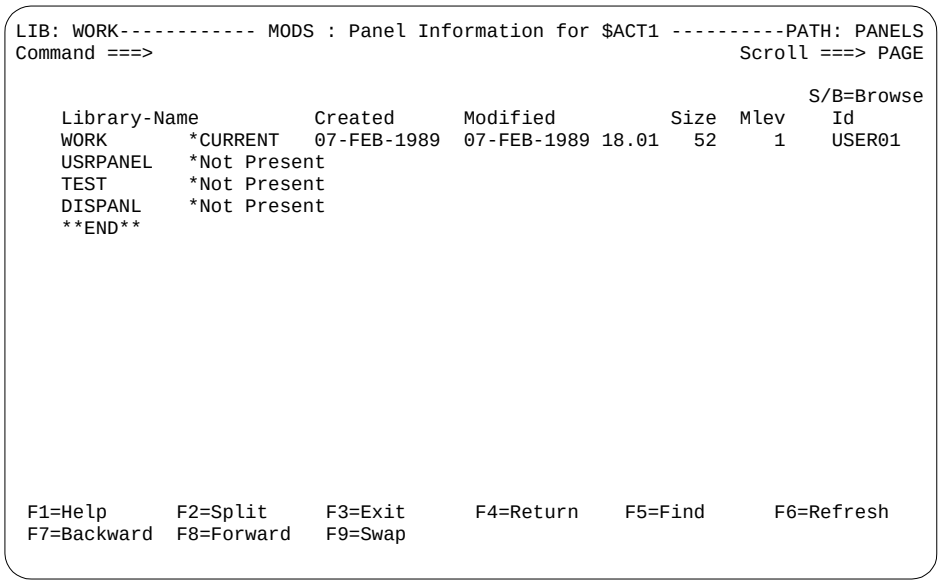
Display Information About a Panel Definition

Select option **I** to display a list of all the libraries, in your path, that contain a specified panel. You must specify the name of the panel in the **Panel Name** field on the menu.

All libraries in your path are displayed (in concatenation order) as shown in Figure 6-5.

The top left of the display shows the library name which was specified on the MODS : Panel Maintenance Menu. This is called the *current* library. The top right of the panel displays the *path name*.

Figure 6-5. MODS : Panel Information Panel



If the panel is not defined in a library, ***Not Present** is displayed next to the library definition. If the panel is present in a library, the following information is displayed:

Library-Name
The identifier of a library. If the panel is present in the current library, ***CURRENT** is displayed next to the library name (and the fields below display relevant library information); otherwise ***Not Present** is displayed.

Created
The date the panel definition was created in this library.

Modified
The date the panel definition was last modified in this library.

Size
The size of the panel definition.

Mlev
The modification level of the panel definition (that is, the number of times the panel has been modified).

Id
The identifier of the person who last updated the panel definition.

Statistics for the panel in the current library are displayed in high intensity. This information tells you where the highest instance of a member is in the library concatenation.

Enter an **S** or **B** next to a library to browse the panel definition stored in that library.

Moving and Copying Panel Definitions Between Libraries

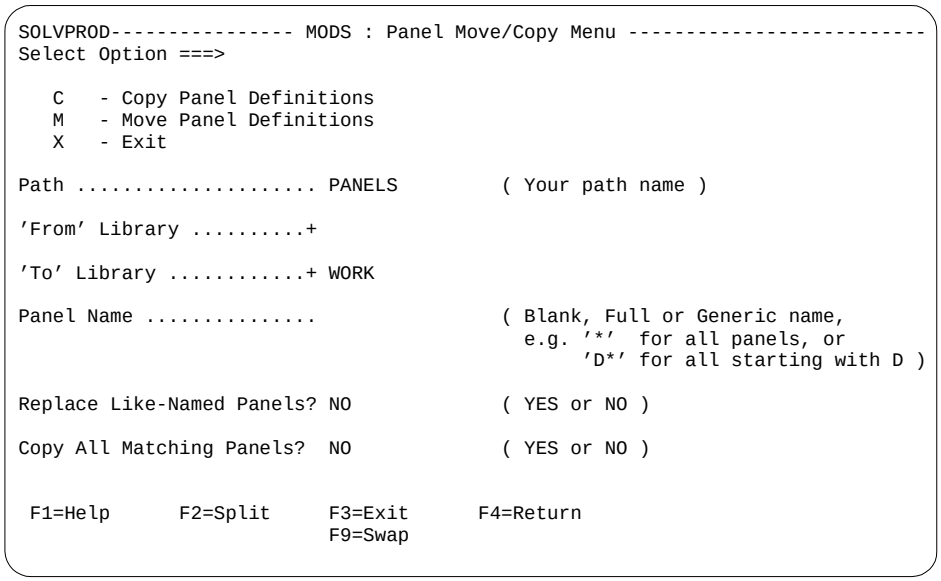
Select option **M** from the MODS : Panel Maintenance Menu to move or copy panels from one library to another within your path.

If you enter a name in the **Library** field or the **Panel Name** field on the MODS : Panel Maintenance Menu, this information is used on the Move/Copy Menu.

Note
To copy panels between libraries on different paths, see Chapter 16, *Maintaining Panel Libraries.*

The MODS : Move/Copy Menu (Figure 6-6) is displayed, allowing you to select the libraries and panels that you want to move or copy.

Figure 6-6. MODS : Panel Move/Copy Menu



When the Copy option is chosen, the definition is loaded from the **FROM Library** and stored into the **TO Library**. When the Move option is chosen, the definition is deleted from the **FROM Library** and added to the **TO Library**.

To Move or Copy a panel, enter the names of the **FROM Library** and **TO Library**. Both libraries must be in your path (your path name is displayed on the screen, and cannot be changed). The **TO Library** must be editable. For Move processing, the **FROM Library** must be editable as well. To select from a list of library names in your path, enter a question mark (?) next to either **Library** field.

Panel Name

Specify the panel name you wish to move or copy in the **Panel Name** field. You can specify the full name to copy a particular panel, or leave the field blank for a selection list of panels in the **FROM Library**.

The panel name can also be specified generically, using the asterisk character (*) as a *wildcard* match for the panel names. An asterisk in the middle of a name matches one character in that position, while one on the end of the name matches any number of characters on the end of a panel name.

For example:

- *ABC* matches the panel named ABC.
- *A*C* matches any panel with a three letter name starting with A and ending with C.
- *ABC** matches any panel whose name starts with the letters ABC.
- * matches every panel in the library.

Specifying a generic name works in conjunction with the specification of the **Copy All Matching Panels?** field (see below).

The following two fields affect the way the Copy or Move operates.

Replace Like-Named Panels?

Valid values are YES and NO. Panels are either added or replaced in the **TO Library** according to the value of this field. Specifying NO safeguards the existing contents of the **TO Library** against unintentional erasure.

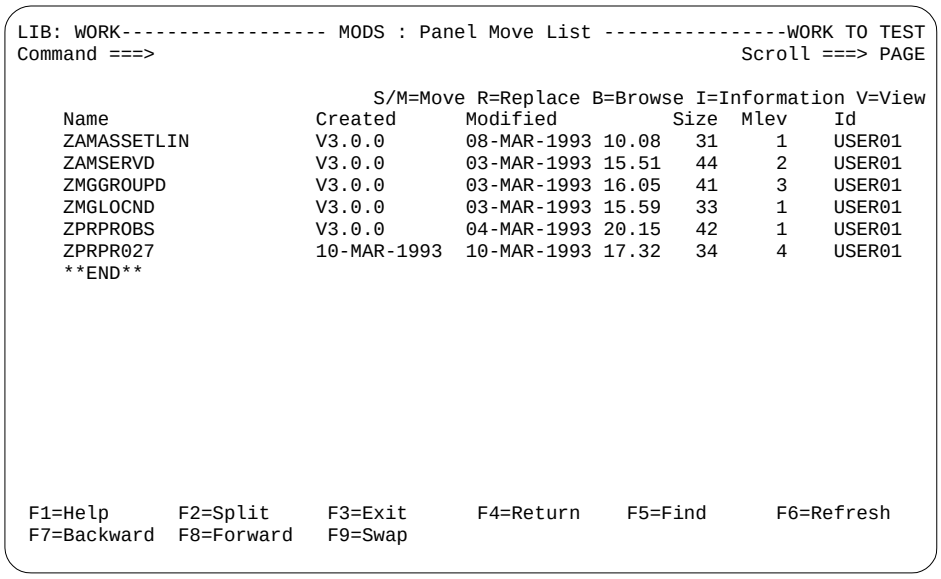
Copy All Matching Panels?

Valid values are YES and NO. Specifies what happens when a generic **Panel Name** is entered (see above). If YES is specified, all panels in the **FROM Library** which match the generic specification are copied (or moved) to the **TO Library** (subject to the value of **Replace Like-Named Panels?**). If NO is specified, a selection list of panels in the **FROM Library** which match the generic name is presented.

Panel Move/Copy List

The MODS : Panel Move/Copy List, as shown in Figure 6-7, is displayed when you select Move or Copy on the MODS : Panel Move/Copy Menu and leave the **Panel Name** blank or specify a generic panel name.

Figure 6-7. MODS : Panel Move/Copy List



The top left of the display shows the name of the library being listed (the **FROM Library**); the top right of the display shows the **FROM Library** and the **TO Library**. The list shows panels in the **FROM Library**, based on the panel name you specified.

You can apply the following actions against members of this list.

Mnemonic	Action
S (or M/C)	Move/Copy the panel (subject to the **Replace Like-Named Panels** field on the Move/Copy menu)
R	Replace the panel—the panel is either Moved or Copied
B	Browse the panel definition
I	Obtain Information about the panel
V	View the panel

Searching Panels for a Character String

Select option **S** from the MODS : Panel Maintenance Menu to search all or specific panels defined in a particular library for a string of characters. Enter the name of the library that contains the panels to be searched in the **Library** field of the MODS : Panel Maintenance Menu.

The MODS : Search Panel Definitions panel (as shown in Figure 6-8) is displayed, allowing you to enter the character string and optionally a panel name prefix.

Figure 6-8. MODS : Search Panel Definitions Panel

```
SOLVPROD-------------- MODS : Search Panel Definitions -----------------------
Command ===>

Search String ............. ____________________________________________________
Panel Name Prefix ......... ____________




 F1=Help      F2=Split     F3=Exit                              F6=Action
                           F9=Swap
```

The fields displayed on the MODS : Search Panel Definitions panel are as follows:

Search String
Specify the string of characters to search for in the panels.

Panel Name Prefix

Use this field to restrict the panels to be searched. Only panels with names that start with the prefix are searched.

The search results in a PSM report that lists the panels containing the specified string of characters. After you fill in the fields, press the ACTION key. The PSM : Confirm Printer panel is displayed.

If necessary, change the values in the fields, then press the CONFIRM key to start the search. If the number of panels to be searched is 100 or more, a panel appears on your screen to advise you of the progress of the search. When the search is complete, a message appears on your screen to advise you the success of failure of the search.

If the report is on hold, you can use the **PQ**[UEUE] command to access the PSM output queue and view the report.

Designing Panels

This section describes what to put in a panel definition: how to define constant data, and input and output fields. A reference section on using the *panel control statements* is included, beginning on page 6-27. The panel control statements can be used to embed comments in panel definitions, to define fields, and to interact with NCL processes.

For details of how NCL processes use panel definitions, see the chapter titled *Designing Interactive Screens (Panel Services)* in the *Network Control Language User's Guide*.

Panel Control Statements

Optional control statements can precede a panel to specify the particular requirements for that panel. These are:

#ALIAS

Defines an alternative name for an input variable.

#OPT

Defines optional operational requirements.

#FLD

Defines or modifies a field character's attributes.

#ERR

Defines the action to be taken for an error condition.

#NOTE

Provides installation documentation (this is ignored during processing).

#TRAILER

Provides a means of placing specified panel lines at the end of the panel (regardless of screen size).

Control statements included within panel definitions must precede the displayable portion of the panel (as determined from the first non-control statement encountered). Control statements must start in column 1 of the lines on which they appear.

Before a panel is displayed, its associated control statements are parsed and any variable substitution performed. This allows control statements to be dynamically tailored.

Data in Panels

Panels contain a combination of fixed data and variable output data:

- Fixed data is the screen captions, field identification text, and other static screen information defined when the panel is created. This does not change when the panel is displayed.
- Variable output data is data generated by the system while the panel is being displayed. It replaces variables positioned within the panel created by the editor. Data is extracted from NCL variables available at the time the panel is invoked.

Variable output data can be displayed in:

- Protected output-only fields (where the data comprises either system or user variables), or
- Unprotected input fields, for any user variables. Once displayed, you can enter data into the unprotected input fields. Panel Services then inserts this data into the user variable for each field, so it is available for further processing by NCL procedures.

The syntax for defining variables within a panel is discussed in the reference section under the heading *#FLD*, on page 6-32.

Panel Design

A panel design contains a series of lines, each of which can contain one or more fields.

Each field is preceded by a field character which identifies the attributes for that field. These attributes specify:

- The field type (input, output, selector pen detectable (SPD), or null)
- The intensity (brightness) of the display
- Optional formatting rules
- Optional editing rules
- The color and extended highlighting used when the field is displayed (for appropriate terminals)

Field Characters

Each panel line has one or more *fields*, starting with a *field character* which specifies the field attributes.

Within the #FLD control statement at the top of the panel definition, you must specify which characters are required for different types of field.

There are two ways of defining field characters:

- *Character mode*—To specify character mode, use any special character other than an alpha or numeric character, and excluding ampersand (&), blank, or null.
- *Hexadecimal mode*—To specify hexadecimal mode, enter the hex value for the character (for example, as X‘FA’). Use any hex value in the range X‘00’ to X‘FF’, excluding the values X‘00’ (null), X‘40’ (blank), X‘50’ (ampersand—&), X‘0E’, and X‘0F’. Hexadecimal mode is used when you need a very large number of field types within one panel and there are insufficient special keyboard characters available to accommodate all of the field characters you require.

Note

If using hexadecimal mode field characters, you must prime the panel definition with the preparse option to assign correct values to field character positions in the actual panel.

Each field is allocated a *field type* which specifies the method for processing the field. Four field types are supported:

OUTPUT
Display only—no data can be entered from the screen.

INPUT
You can both display and enter data.

SPD
Selector pen detectable—data cannot be typed in.

NULL
Display only—although unprotected, any data entered is ignored.

Any mixture of the above field types can be defined to suit the requirements for a panel you are designing.

The field character that precedes each field determines:

- The field type
- The display characteristics of the field (such as intensity, color, highlighting, justification, and capitalization)
- For input fields, the internal validation rules that must be obeyed for data entered in that field. Such rules can specify, for example, that a field is mandatory, must be numeric, cannot contain imbedded blanks, or must be a valid date.

Each field character that you define occupies the equivalent screen position when the panel is displayed, but appears as a blank character (the *attribute byte*).

The field proper starts from the next position after the field character, and continues to the next attribute byte on the same line, or to the end of that line where there is no intervening field. Fields do not wrap round from one line to the next.

Field characters can be specified either in *character*, in which case they are always special characters (non-alpha, and non-numeric; for example, *, %), or in hexadecimal.

The three standard default field characters are:

%
High-intensity, protected (no input)

+
Low-intensity, protected (no input)

–
High-intensity, unprotected (input, no validation)

The above standard default field characters do not require definition by a #FLD statement.

Define any additional field characters you need using the #FLD statement. The attributes for the above default field characters can be modified. You can use the #OPT statement to nominate alternative standard field characters, so that %, +, and _ can be used within the panel and not processed as field characters, if required.

Column 1 of each line of a panel must be a valid field character; if one is not defined, then the attributes for the second standard field character (normally +, for low-intensity, protected) are used to replace any data incorrectly placed in that column.

In Figure 6-9, all fields preceded by a percent sign (%) display in high-intensity and are protected from data entry. All fields preceded by a plus sign (+) display in low-intensity and are also protected. The only field available for input is on line 16 of the text data, preceded by an underline (_). The word ***newpanel*** identifies the NCL variable that receives the data that the user enters in this field once the ENTER key is pressed.

By default, the cursor is placed at the beginning of the ***cmd*** field, as this is the first field requiring input—no other cursor position has been specified.

Note

The ampersand (&) which precedes a variable is omitted when specifying an input field.

Figure 6-9. Sample Panel Using Default Field Characters

```
LIB: USRPANEL ---------------- MODS : Edit Panel -----------CAPS OFF--NULLS ON-
COMMAND ===>                                    NAME: TESTPANEL1   SCROLL: HALF
      ***TOP-OF-DATA***                                            L1 C1 74
''''                              %MODS : Rename Panel
'''' +Command ===>_cmd
''''
''''
''''
'''' %   Current Panel Definition
''''
'''' +   Path ...................%&path      +
''''
'''' +   Library ................%&lib  +
''''
'''' +   Panel Name ..............%&oldpanel +
''''
'''' %   Enter New Panel Name
''''
'''' +   New Panel Name ..........._newpanel +
''''
''''  ***END-OF-DATA***
```

When a panel is displayed, field characters are removed and the required terminal attribute characters substituted. Figure 6-10 shows how the panel in Figure 6-9 appears when displayed.

Note

In all figures, the underline symbol (_) designates the cursor location.

Figure 6-10. Sample Panel Display

```
                              MODS : Rename Panel
Command ===> _


   Current Panel Definition

   Path ...................._________________

   Library ................._________

   Panel Name ..............___________

   Enter New Panel Name

   New Panel Name ..........________
```

Allowing Long Field Names in Short Fields

An input field is defined on a panel by inserting an appropriate attribute character followed by the name of the NCL variable which contains the input data. Unfortunately this means that input fields cannot be any shorter than the variable name which contains the input data.

The #ALIAS control statement allows you to define an alias name for a variable. The alias can be used where the variable would have been used. A range or list of variables can be defined and referred to by the same alias name in the panel definition.

Output Padding and Justification

Careful use of padding and justification greatly enhances the look and effectiveness of panels for end-users.

Panel Services includes extensive facilities to manipulate displayed data. Padding and justification qualities are specified by the #FLD statement. There are two justification categories—field level justification, and variable level justification. Both can be used concurrently.

Field Level Justification

This is performed on an entire field as delimited by defined field characters. Field justification analyzes the entire field, strips trailing blanks, and pads and justifies the remaining data. The #FLD operands controlling field level justification are JUST and PAD.

The various ways data can be manipulated are best described by a series of examples. These examples show a mix of fields each defined with a different field character and each showing a different display format. Study the #FLD statements and observe the results achieved.

```
#NOTE      This sample panel definition gives examples of the
#NOTE      use of field level justification and padding.
#FLD       #
#FLD       $     JUST=RIGHT
#FLD       @     JUST=LEFT PAD=<
#FLD       ?     JUST=RIGHT PAD=>
#FLD       /     JUST=CENTER PAD=.
#&VAR01          +
@&VAR03          +
?&VAR04          +
/&VAR05          +
```

Assume the following variable assignment statements are executed by the NCL procedure before displaying the sample panel:

```
&VAR01 = &STR Left justified null padding
&VAR02 = &STR Right justified null padding
&VAR03 = &STR Left justified with padding
&VAR04 = &STR Right justified with padding
&VAR05 = &STR Center justified with padding
```

The default values are JUST=LEFT and PAD=NULL, as shown by the first line in the example below, where field character # is used with no attributes other than the defaults. The sample panel is displayed as follows:

```
Left justified null padding
                                    Right justified null padding
Left justified with padding<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Right justified with padding
..............Center justified with padding..............
```

Variable Level Justification

This operates independently of field level justification, and applies to the data substituted for field variables defined as requiring variable level justification. Variable level justification is designed to help tabulated output, where data of differing lengths is substituted for a series of variables and where the normal substitution process disrupts display formats. The #FLD operands that control variable level justification are VALIGN and PAD.

The substitution process substitutes data in place of the &variable without creating additional characters. Thus, if a variable (for instance, &VARIABLE) is replaced by data (for example, *Data*), any characters following this are moved left to occupy any spaces remaining after substitution (this occurs if spaces are freed going from a long variable name to a shorter data length).

```
#NOTE   This sample panel definition gives examples of the
#NOTE   use of variable justification, padding, and field
#NOTE   justification.
#FLD # VALIGN=LEFT
#FLD $ VALIGN=RIGHT
#FLD @ VALIGN=CENTER
#FLD ? VALIGN=LEFT PAD=.
#FLD / VALIGN=RIGHT PAD=.
#FLD } VALIGN=CENTER PAD=.
#FLD ! VALIGN=LEFT JUST=RIGHT PAD=.
#&VARIABLE other data+
$&VARIABLE other data+
@&VARIABLE other data+
?&VARIABLE other data+
/&VARIABLE other data+
}&VARIABLE other data+
!&VARIABLE other data+
```

Variable level justification, controlled by the VALIGN operand of the #FLD statement, lets you influence the way substitution is performed.

Note

Variable level justification is only performed if the length of the data being substituted is less than the length of the variable name being replaced, including the ampersand (&).

Assume the following variable assignment statement has been executed by the NCL procedure before displaying the sample panel:

```
&VARIABLE = Data
```

&VARIABLE is the only variable within a field which contains the words ‘other data’. Where both field justification and variable alignment are used, the padding character applies to both, as shown by the last line of the example for the field character !. The sample panel is displayed as follows:

```
Data    other data
Data other data
Data  other data
Data..... other data
.....Data other data
..Data... other data
................................Data..... other data
```

Input Padding and Justification

Fields to which the PAD and JUST operands of the #FLD statement are applied can be defined as input fields. If an input field is primed with data during the display process, the alignment of data within that field when displayed is as described above in the section on *Output Padding and Justification*, except that JUST=CENTER is treated as JUST=LEFT. When Panel Services processes input from the screen, input fields defined using the PAD and JUST operands are specially processed using the following rules:

- Trailing blanks and pad characters are stripped off, unless the pad character is numeric.
- If JUST=RIGHT is specified for the field, leading blanks and pads are stripped off (including numeric pads).
- If JUST=ASIS is specified for the field, trailing blanks and pads are stripped off, but leading blanks and pads remain intact.

```
#NOTE   This sample panel definition gives examples of the
#NOTE   use of input padding and justification.
#FLD # TYPE=INPUT
#FLD $ TYPE=INPUT JUST=RIGHT
#FLD @ TYPE=INPUT PAD=< JUST=LEFT
#FLD ? TYPE=INPUT PAD=> JUST=RIGHT
#FLD / TYPE=INPUT PAD=0 JUST=LEFT
#FLD } TYPE=INPUT PAD=1 JUST=RIGHT
#VAR01                                +
$VAR02                                +
@VAR03                                +
?VAR04                                +
/VAR05                                +
}VAR06                                +
```

Assume the following variable assignment statements are executed by the NCL procedure before displaying the sample panel:

```
&VAR01 = Walt
&VAR02 = Tom
&VAR03 = Dick
&VAR04 = Harry
&VAR05 = John
&VAR06 = Vicky
```

The sample panel is displayed as:

```
WALT
                                              TOM
DICK<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>HARRY
JOHN00000000000000000000000000000000000000000000
11111111111111111111111111111111111111VICKY
```

If control is passed back to the NCL procedure without any data entered into the input fields, the variables are set to the following values:

```
&VAR01 = WALT
&VAR02 = TOM
&VAR03 = DICK
&VAR04 = HARRY
&VAR05 = JOHN
&VAR06 = VICKY
```

Note

If the line JOHN000... is modified, it is padded to the right with zeros.

The variable values are translated to uppercase because the default for input fields is CAPS=YES.

Displaying Function Key Prompts

The SAA Common User Access (CUA) standards require that a list of function keys and their functions be displayed at the bottom of a panel. These function key prompts are displayed on the last lines of the physical screen.

The #TRAILER control statement can be used by NCL procedures to nominate lines which appear at the bottom of the panel. The function key prompts are then always displayed at the bottom of the panel regardless of the screen size. The #TRAILER control statement is described below.

#ALIAS

Function:

Defines an alternative name for input variables.

```
#ALIAS        name
             {  VARS=prefix*[ RANGE=(start,end) ]  |
                VARS={ vname | (vname,vname, ...vname) } }
```

Use:

To allow the panel definition to contain alternative names for variable names in **TYPE=INPUT** and **TYPE=OUTVAR** fields.

This facility is useful if you require short fields with long variable names. Each reference to *name* in the panel definition is regarded as a reference to the next name from the list of VARS specified.

Operands:

name

Specifies the alias name which appears in the panel definition. Whenever this name occurs in a field declared as **TYPE=INPUT** or **TYPE=OUTVAR** on the #FLD statement, panel services logically replaces it with the next available name from the VARS list.

The name can be from one to eight characters in length. The first character must be an alphabetic or national character. The remaining characters, if any, must be alphanumeric or national characters.

The same name can be used on multiple #ALIAS statements. The variable names are simply added to the end of the list of names to which the alias name refers.

VARS=*prefix [RANGE=(*start,end*)] |**
VARS=(*vname,vname, ..., vname*) }

Specifies the list of names which replaces the alias name in the panel definition. Each time the alias name is encountered in the panel definition it is replaced by the next available name from this list. The format of the operands associated with VARS= is:

VARS=*prefix** denotes that the variable names used are *prefix*1, *prefix*2, and so on. The RANGE= operand can be specified to indicate a starting and ending suffix number. The default is RANGE=(1,64). The format *prefix** cannot be used in conjunction with other variable names on the same #ALIAS statement.

VARS=*vname* is the name of a variable excluding the ampersand (&).

Examples:

```
#ALIAS Z VARS=LONGNAME
#ALIAS Z123 VARS=(SATURDAY,SUNDAY)
#ALIAS AVAR VARS=LINE* RANGE=(10,20)
```

Notes:

Multiple #ALIAS statements can be used for the same alias name if insufficient space is available on a single statement**.**

If an alias name appears in the panel definition after all the variable names in the alias list have been used up, the alias name itself appears in the panel.

Symbolic variables can be included in the #ALIAS statement. Variable substitution is performed prior to processing the statement, using variables available to the NCL procedure at the time the &PANEL statement is issued.

See Also:

The #FLD panel control statements.

#ERR

Function:

Defines action to be taken during error processing.

```
#ERR    [  INTENS={ HIGH | LOW  } ]
        [  { COLOR | COLOUR } = { BLUE | RED | PINK | GREEN |
                        TURQUOISE | YELLOW | WHITE | DEFAULT }  ]
        [  { HLIGHT | HLITE } = { USCORE | REVERSE | BLINK | NONE }  ]
        [  ALARM={ YES | NO  } ]
```

Use:

The #ERR statement is a panel control statement which determines the processing required when a panel is being redisplayed following an error condition.

An error condition can be detected either by Panel Services internal validation or by the processing NCL procedure. If detected by internal validation, (and &CONTROL PANELRC is not in effect), error processing is automatically invoked by Panel Services. If detected by the processing NCL procedure, error processing is invoked in one of two ways:

- Using the &ASSIGN OPT=SETERR verb
- By nominating the name of the variable that identifies the invalid input field on the ERRFLD operand of the #OPT statement. This is dynamically invoked by using a symbolic variable with the ERRFLD operand and setting this variable to the name of the variable (minus the ampersand) that identifies the field in error.

See the section, *Handling Errors,* in the Panel Services chapter in the *Network Control Language User's Guide,* for a detailed discussion on using this technique.

When #ERR processing is initiated, the cursor is positioned to the first field in error and the panel is redisplayed, applying the attributes defined on the #ERR statement to the fields in error. This technique provides the panel user with a simple means of drawing the terminal operator's attention to the field in error. This is particularly effective on color terminals where the color of any field in error can be altered for the duration of the error, and reverts to normal when the error condition is rectified.

One or more #ERR statements can be defined in any order. However, as with #OPT, #FLD, and #NOTE statements, any #ERR statement must precede the start of the panel, which is determined by the first line that is not a control statement.

Operands:

INTENS={ HIGH | LOW }

Determines the intensity of the error field when displayed. The INTENS operand is ignored for terminals with extended color and highlighting when either the COLOR or HLIGHT operands are specified.

HIGH specifies that the field is displayed in double intensity.

LOW specifies that the field is displayed in low or standard intensity.

{ COLOR | COLOUR } = { BLUE | RED | PINK | GREEN | TURQUOISE | YELLOW | WHITE| DEFAULT }

Determines the color of the field. It applies only to IBM terminals with seven-color support and Fujitsu terminals with three- or seven-color support.

The COLOR operand is ignored if the terminal does not support extended color. This enables COLOR to be specified on panels that are displayed on both color and non-color terminals. COLOR can be used in conjunction with the HLIGHT operand.

For Fujitsu terminals that support extended color datastreams, but support only three colors, the following color relationships are used:

Specified	**Result** (on Fujitsu three-color terminal)
GREEN	GREEN
RED	RED
PINK	RED
BLUE	GREEN
TURQUOISE	GREEN
YELLOW	WHITE
WHITE	WHITE
DEFAULT	GREEN

Fujitsu seven-color terminals are treated the same as IBM seven-color terminals.

The DEFAULT keyword indicates that the color of the field is determined from the INTENS operand. This is particularly useful if you want to set the color from an NCL procedure (that is, COLOR=&COLOR is specified and the NCL procedure can set the &COLOR variable to DEFAULT).

{ HLIGHT | HLITE } = { USCORE | REVERSE | BLINK | NONE }

Applies only to terminals with extended highlighting support, and determines the highlighting to be used for the field.

Because the HLIGHT operand is ignored if the terminal does not support extended highlighting, HLIGHT can be specified on panels that are displayed on terminals that do not support extended highlighting. HLIGHT can be used with the COLOR operand.

When NONE is specified, the HLIGHT operand is ignored and no extended highlighting is performed for this field.

ALARM={ YES | <u>NO</u> }

Turns the terminal alarm on or off if the panel is displayed with an error condition. This works independently of the ALARM operand on the #OPT control statement.

Examples:

```
#ERR COLOR=RED HLIGHT=REVERSE ALARM=YES
#ERR COLOR=YELLOW HLIGHT=BLINK INTENS=HIGH
```

Notes:

Only those attributes defined on the #ERR statement are modified for the field in error. All other attributes associated with the original field, such as internal validation, remain intact

Symbolic variables can be included in a #ERR statement. Variable substitution is performed before processing the statement, by using variables available to the NCL procedure at the time the &PANEL statement is issued.

When &CONTROL PANELRC is in effect, internal validation does not automatically reshow the panel with the error message, and so on. In this case, the procedure regains control following the &PANEL statement with the &RETCODE system variable set to 8 to indicate that an error has occurred. The &SYSMSG variable contains the text of the error message that describes the error and the &SYSFLD variable contains the name of the field in error. This name is the name of the variable in an input field that receives the data entered into that field.

The &ASSIGN statement SETERR option provides a mechanism for assigning #ERR field attributes to multiple (input field) variables before displaying a panel. This allows you to accept input from a number of different fields on a panel, validate all the fields and then redisplay the panel with all incorrect fields displayed with the #ERR attributes. This shows the user all the errors at one time, rather than field by field.

See Also:

The #OPT, #FLD, and #NOTE panel control statements and the &ASSIGN statement.

#FLD

Function:

Defines or modifies a panel definition field character.

```
#FLD    {  c | X'xx' }
        [  BLANKS={ TRAIL | NONE | ANY } ]
        [  CAPS={ YES | NO } ]
        [  { COLOR | COLOUR } = {  BLUE | RED | PINK | GREEN |
                   TURQUOISE | YELLOW | WHITE | DEFAULT } ]
        [  CSET={ ALT | DEFAULT } ]
        [  EDIT={ ALPHA | ALPHANUM | DATEn | DSN | HEX |
                   NAME | NAME* | NUM | REAL | SIGNNUM | TIMEn }  ]
        [  { HLIGHT | HLITE } = { USCORE | REVERSE | BLINK | NONE }  ]
        [  INTENS={ HIGH | LOW | NON } ]
        [  JUST={ LEFT | RIGHT | ASIS | CENTER | CENTRE } ]
        [  MODE={ SBCS | MIXED } ]
        [  NCLKEYWD={ YES | NO } ]
        [  OUTLINE={  {L R T B} | BOX } ]
        [  PAD={ NULL | BLANK | char } ]
        [  PSKIP={ NO | PMENU } ]
        [  RANGE=(min,max) ]
        [  REQUIRED={ YES | NO } ]
        [  SKIP={ YES | NO } ]
        [  SUB={ YES | NO } ]
        [  TYPE={ OUTPUT | INPUT | OUTVAR | SPD | NULL } ]
        [  VALIGN={ NO | LEFT | RIGHT | CENTER | CENTRE } ]
```

Use:

The #FLD statement is a panel control statement used to tailor the operational characteristics of a panel.

When a panel is defined it is constructed of a number of lines which in turn are made up of a number of fields. Each field commences with a field character which appears as a blank on the panel when displayed. Each field character determines the attributes that are to be associated with the field following the field character itself. A field is delimited by the next field character or the end of the panel line. Fields cannot wrap from one line to the next.

The first field on a line always starts in column 1. If no field character is defined in the first position of the line, the attributes of the second of the three standard field characters (usually a plus sign (+), TYPE=OUTPUT, INTENS=LOW) are forced and replace any non-field character incorrectly placed in this position.

Before parsing, the #FLD statement is scanned and variable substitution is performed. This makes it possible to dynamically tailor any of the options or operands on the statement.

As many #FLD statements as required can be specified. They can be defined in any order. However, as with #OPT, #ERR and #NOTE statements, all #FLD statements must precede the start of the panel, which is determined by the first line that is not a control statement.

Operands:

***c* | X'*xx*'**

The field character:

c is the character that is used in the panel definition to identify the start of the field. This is known as a *field character.* This must be a single non-alpha and non-numeric character. Any special character (for example, an exclamation mark) can be used, with the exception of an ampersand (&), which is reserved for use with variables.

X'*xx*' is the *hexadecimal* value of the field character. Use this notation to specify any value in the range X'01' to X'3F'. Do not use values which correspond to alphanumeric characters , or X'0E' (shift in) or X'0F'(shift out).

Although the panel editor does not allow you to enter non-displayable hex attributes (X'01' to X'3F') in the body of the panel, you can use the preparse option to prime the field character value in the panel before issuing the &PANEL statement.

The first #FLD statement to reference a particular field character defines a new character. Subsequent statements referencing that same field character modify or extend the attributes of the field character. Three standard field characters (%, +, _ unless altered by the #OPT statement) are provided. If a default field character (usually % + or _) is referenced, it is the same as extending or modifying the attributes of an existing field character.

If no additional operands are defined following a new field character, the following defaults apply:

TYPE=OUTPUT INTENS=LOW

No special attributes or internal validation apply.

BLANKS={ TRAIL | NONE | <u>ANY</u> }

This determines the format for entering data in input fields. By default, a field can contain imbedded blanks (BLANKS=ANY). Specification of this operand ensures that the entered data does not contain imbedded blanks and contains only trailing blanks (TRAIL) or no blanks at all (NONE). This operand works independently of the REQUIRED operand. For optional fields this operand can still be specified to ensure that when data is entered, it is in the correct format. If &CONTROL PANELRC is not in effect, BLANKS=TRAIL is specified, and the data is in error, Panel Services redisplays the panel with the &SYSMSG variable set to:

INVALID IMBEDDED BLANKS

If BLANKS=NONE is specified and the data is in error, Panel Services redisplays the panel with the &SYSMSG variable set to:

INCOMPLETE FIELD

If &CONTROL PANELRC is in effect, control is returned to the NCL procedure for error handling instead of being handled totally by Panel Services. In this case, &SYSFLD contains the name of the field in error and &SYSMSG the error message text.

CAPS={ <u>YES</u> | NO }

Applies to input fields only and determines if entered data is converted to upper case before passing it to the NCL procedure in the nominated variable. Conversion to upper case is also performed for the data associated with an input variable before displaying the panel. This does not impact the current contents of the variable unless the data is modified and entered by the operator. Output fields are displayed exactly as defined and are not subject to upper case conversion.

Note

The effect of CAPS=NO can be negated if the variable that receives the data is used in an assignment statement (for example, &A = &DATA) within the processing NCL procedure, as data can be converted to upper case before performing the assignment; see the &CONTROL UCASE option.

The CAPS operand is ignored when operating in a system executing with SYSPARMS DBCS=YES.

{ COLOR | COLOUR } = { BLUE | RED | PINK | GREEN | TURQUOISE | YELLOW | WHITE | DEFAULT }

Applies only to IBM terminals with seven-color support and Fujitsu terminals with three- or seven-color support, and determines the color of the field.

Because the COLOR operand is ignored if the terminal does not support extended color, COLOR can be specified on panels that are displayed on both color and non-color terminals. COLOR can be used in conjunction with the HLIGHT operand.

For Fujitsu terminals that support extended color datastreams where only three colors are available, the following color relationships are used:

Specified	**Result** (on Fujitsu three-color terminals)
GREEN	GREEN
RED	RED
PINK	RED
BLUE	GREEN
TURQUOISE	GREEN
YELLOW	WHITE
WHITE	WHITE
DEFAULT	GREEN

Fujitsu seven-color terminals are treated the same as IBM seven-color terminals.

The DEFAULT keyword indicates that the color of the field is determined from the INTENS operand. This is particularly useful if you want to set the color from an NCL procedure (that is, COLOR=&COLOR is specified and the NCL procedure can set the &COLOR variable to DEFAULT).

CSET={ ALT | <u>DEFAULT</u> }

Applies to output fields only. The operand determines which terminal character set to use to display the field. If you specify CSET=ALT (or ALTERNATE), you can draw box shapes using the following characters:

e	is displayed as	
s	is displayed as	
D	is displayed as	
E	is displayed as	
M	is displayed as	
N	is displayed as	
F	is displayed as	
G	is displayed as	
O	is displayed as	
P	is displayed as	
L	is displayed as	

Note

CSET=ALT supersedes CSET=ASM in version 3.1

EDIT={ ALPHA | ALPHANUM | DATE*n* | DSN | HEX | NAME | NAME* | NUM | REAL | SIGNNUM | TIME*n* }

For input fields this determines additional internal editing to be performed by Panel Services. By default no editing is performed. Specification of this operand ensures that the entered data conforms to the nominated type. If a field is mandatory, then REQ=YES should also be specified.

ALPHA

Only accept A to Z

ALPHANUM

Only accept A to Z, 0 to 9, #, @ and $

DATE*n*

Field must be a valid date. The DATE*n* keyword must correspond to one of the &DATE*n* system variables, and indicates that the input field must contain date in the format associated with that system variable. For example, EDIT=DATE5 indicates that the input field must always contain a date in the format corresponding to the &DATE5 system variable (MM/DD/YY).

DSN

Field must be a valid OS/VS format dataset name. If required, a partitioned dataset (PDS) member name or Generation Data Group (GDG) number can be specified in brackets as part of the name.

HEX

Only accept 0 to 9 and A to F.

NAME

Field must commence with alpha (A to Z, #, @ or $) and be followed by alphanumerics (A to Z, 0 to 9, #, @ or $).

NAME*

Field must commence with alpha (A to Z, #, @ or $) and be followed by alphanumerics (A to Z, 0 to 9, #, @ or $) but can be terminated with a single asterisk (*). This allows a value to be entered that can be interpreted as a generic request by the receiving procedure.

NUM

Only accept 0 to 9. Specifying EDIT=NUM, in addition to internal Edit validity checking, sets a hardware flag to inhibit alpha input. This flag is display system dependent for its implementation; either in hardware or emulation software.

REAL

Input in this field must conform to the syntax for integers, signed numbers or real numbers, including scientific notation. Refer to the chapter titled *Arithmetic in NCL* in the *Network Control Language User's Guide* for information on real number support.

SIGNNUM

Field must be numeric but can have a leading sign (+ or -).

TIME*n*

Field must be a valid time. The TIME*n* keyword must correspond to one of the &ZTIME*n* system variables and indicates that the input field must be in the format associated with that system variable.

When invalid data is detected and &CONTROL PANELRC is not in effect, standard error processing is invoked by Panel Services and control is not returned to the NCL procedure until the error is corrected.

For EDIT=NUM the panel is redisplayed with the &SYSMSG variable set to:

FIELD NOT NUMERIC

For EDIT=REAL the panel is redisplayed with the message

FIELD NOT REAL NUMBER

For EDIT=ALPHA, ALPHANUM, HEX or NAME the panel is redisplayed with the &SYSMSG variable set to:

INVALID VALUE

For EDIT=DATE*n* the panel is redisplayed with the &SYSMSG variable set to:

INVALID DATE

For EDIT=DSN the panel is redisplayed with the &SYSMSG variable set to:

INVALID DATASET NAME or INVALID MEMBER NAME

For EDIT=TIME*n* the panel is redisplayed with the &SYSMSG variable set to:

INVALID TIME

In all cases the terminal alarm sounds and the cursor is positioned to the field in error. If a #ERR statement has been included in the panel definition, processing of the error condition is performed as defined in that statement.

If &CONTROL PANELRC is in effect, control is returned to the NCL procedure for error handling instead of being handled totally by Panel Services. In this case, &SYSFLD contains the name of the field in error and &SYSMSG the error message text.

Note

Use of the EDIT operand might also require the use of the BLANKS operand to ensure that entered data does not include imbedded blanks. Regardless, editing is performed only for the length of the data entered and not for the length of the input field. If the entire field must be entered, the BLANKS=NONE operand should be specified.

{ HLIGHT | HLITE } = { USCORE | REVERSE | BLINK | <u>NONE</u> }

Applies only to terminals with extended highlighting support, and determines the highlighting to be used for the field.

Because the HLIGHT operand is ignored if the terminal does not support extended highlighting, HLIGHT can be specified on panels that are displayed on terminals that do not support extended highlighting. HLIGHT can be used with the COLOR operand.

The NONE keyword is provided as a no-impact value that can be used when the highlighting of a field is being dynamically determined from the NCL procedure and set using variable substitution of the #FLD statement. When NONE is specified, the HLIGHT operand is ignored.

INTENS={ HIGH | LOW | NON }

Determines the intensity of the field when displayed.

HIGH

The field is displayed in double intensity. High intensity is normally associated with input fields and other important data and its use minimized to maintain its effectiveness.

LOW

The field is displayed in low or standard intensity.

NON

The field is displayed in zero intensity. Any data within the field is not visible to the operator. This is normally used for input fields where sensitive data such as passwords are entered. Use of this attribute for output fields is meaningless. Color or extended high-lighting attributes are ignored when used in conjunction with this attribute.

JUST={ LEFT | RIGHT | ASIS | CENTER | CENTRE }

For output fields, this determines the alignment of the data within the field after trailing blanks have been stripped. Justification is applied at a field level and should not be confused with VALIGN which applies to the individual variable only:

- LEFT results in padding to the right
- RIGHT results in padding to the left
- ASIS is treated as JUST=LEFT for output fields
- CENTER results in padding to both the left and the right.

For input fields justification occurs both when the data is being displayed and when the data is being processed on subsequent entry. When an input field is formatted for display (the value currently assigned to the variable defined in the input field is substituted in place of the variable's name) the data is justified to the left and padded to the right for JUST=LEFT or justified to the right and padded to the left for JUST=RIGHT. JUST=CENTER is treated like JUST=LEFT. For JUST=ASIS data is positioned exactly as defined in the variable and padding to the right is performed.

On subsequent re-entry, trailing blanks and pad characters are stripped, unless the trailing pad character is a numeric, in which case it is not stripped:

- For JUST=RIGHT leading blanks and pads are also stripped (including numerics). Use of JUST=RIGHT for input fields can inconvenience terminal operators, as it is necessary to move the cursor to the commencement of the data in the field.
- For JUST=ASIS trailing blanks and pads are stripped, but leading blanks and pads remain intact.

MODE={ SBCS | MIXED }

Applies to IBM terminals capable of supporting DBCS datatreams. If a panel is sent to such a device, input fields on the panel that use this #FLD character allow the operator to enter DBCS characters if MODE=MIXED is specified. IBM DBCS terminals do not allow DBCS character entry in input fields that specify MODE=SBCS (single byte character stream).

Note

This operand does not apply to Fujitsu terminals, which allow DBCS character entry at any time.

NCLKEYWD={ YES | NO }

Specifies whether fields that use this FLD character accept input of words that conflict with NCL keywords. The default is YES. NO causes an attempt to enter any NCL reserved keyword to be rejected.

OUTLINE={ { L R T B } | BOX }

Specifies the extended highlighting outlining option required for this field. Any combination of L (left) R (right) T (top) or B (bottom) can be coded. The field is outlined at the top or bottom with a horizontal line and at the left and right border with a vertical line according to the options specified. Alternatively the BOX option can be specified which is equivalent to specifying LRTB. This option is terminal dependent.

PAD={ NULL | BLANK | *char* }

Applies to INPUT, OUTPUT, and SPD fields.

For output fields PAD works in conjunction with both the JUST and VALIGN operands, one of which must be specified for PAD to take effect. Determines the pad or fill character to be used when the field is displayed.

The variable substitution process substitutes the data currently assigned to any variables within the field being processed. Having completed substitution, any difference between the length of the field defined on the panel and the length after substitution (after stripping trailing blanks) is padded with the specified PAD character.

For input fields, use of the NULL character ensures that the terminal operator can use keyboard insert mode when entering data. Padding is performed either to the left or the right as specified in the JUST operand.

char

specifies a single character that is to be the pad character (for example, PAD=-). There is no restriction on the character used including the use of any of the field characters defined on #FLD statements. Care should be taken when using numeric pad characters as their use impacts the pad character stripping process on subsequent entry.

Note

Using PAD characters with input fields invokes special processing on subsequent input to ensure that unnecessary pad characters are stripped before returning the entered data in the nominated variable. See the section, *Input Padding and Justification*, on page 6-25, for detailed information.

PSKIP={ NO | PMENU }

Applies to input fields only and determines if panel skip requests are accepted in this field. A panel skip request is entered in an input field in the format =*m.m*, where *m.m* is a menu selection. When entered in an appropriate field a panel skip to the specified menu selection is performed.

NO

The input field is not scanned for panel skip requests.

PMENU

The input field is scanned for panel skip requests and actioned.

RANGE=(*min*,*max*)

For numeric fields, specifies the range of acceptable values. The range includes all numbers, from the minimum number (*min*) to the maximum number (*max*). Both *min* and *max* must be specified and *max* must be equal to or greater than *min*. Use of this operand forces EDIT=NUM. If the entered number falls outside the acceptable range, and &CONTROL PANELRC is not in effect, Panel Services redisplays the panel with the &SYSMSG variable set to:

```
NOT WITHIN RANGE
```

If &CONTROL PANELRC is in effect, control is returned to the NCL procedure for error handling instead of being handled totally by Panel Services. In this case, &SYSFLD contains the name of the field in error and &SYSMSG the error message text.

REQUIRED={ YES | NO }

Specifies that this is a mandatory field that must be entered by the user. If &CONTROL PANELRC is not in effect, Panel services rejects any entry by the user unless this field has been entered. If not entered, Panel Services redisplays the panel with the &SYSMSG variable set to:

```
REQUIRED FIELD OMITTED
```

The terminal alarm sounds and the cursor is positioned to the omitted field. If a #ERR statement has been included in the panel definition, processing of the error condition is performed as defined by the #ERR statement. Failure to include the &SYSMSG variable on the panel suppresses this error message. This operand can be abbreviated to REQ=.

If &CONTROL PANELRC is in effect, control is returned to the NCL procedure for error handling instead of being handled totally by Panel Services. In this case, &SYSFLD contains the name of the field in error and &SYSMSG the error message text.

SKIP={ YES | NO }

For output fields only, determines if the skip attribute is to be assigned to the field. The result of using this option is that the cursor skips to the next input field if the preceding input field is entered in full and the intervening output field is specified with the SKIP operand.

Note

This operand is NO by default, as field skipping can unexpectedly place the cursor in the wrong screen window when operating in split screen mode.

SUB={ YES | NO }

For output fields only, determines if variable substitution is to be performed. This operand can be used for fields where data contains the & character. This results in the current value of that variable being substituted, or the variable being eliminated if no value was assigned. This operand is ignored for both INPUT and SPD fields.

TYPE={ OUTPUT | INPUT | OUTVAR | SPD | NULL }

Determines if the field is to be processed as an output-only field (OUTPUT and OUTVAR), input field (INPUT), Selector Pen Detectable (SPD) field or pseudo input field (NULL).

OUTPUT

A protected field is created which does not allow keyboard entry. This field can contain a mixture of fixed data and variables. Each variable must commence with an ampersand (&). Substitution of variables is performed using the variables available to the invoking NCL procedure at the time the &PANEL statement is issued. Global variables can be referenced in an output field. Alignment and padding is performed according to the rules defined for the field.

INPUT

An unprotected field is created that allows keyboard entry. This field must contain a single variable name (without the ampersand). This single variable must immediately follow the field character. System variables and global variables cannot be used in an input field. Subsequent data entered into this field is made available to the invoking NCL procedure in this variable on return from the &PANEL statement. Specification of multiple variables or a mixture of variables and fixed data in an input field results in an error.

OUTVAR

The same as TYPE=OUTPUT except that an & is inserted by Panel Services between the field attribute and the next character. This means that you can follow a TYPE=OUTVAR field character with a variable name without the ampersand. This facility makes it easy to create fields which switch between input and output under NCL control. For example, a panel could contain the statements:

```
#FLD $ TYPE=&INOUT
+ Record Key .....$RKEY +
```

An NCL procedure then sets the variable &INOUT to control if the data in the variable &RKEY is output only, or if it can be modified by an operator:

```
&INOUT = OUTVAR   -* the value is output only.
&INOUT = INPUT    -* the Operator can modify the
                  -* field.
```

Note

A similar effect can be achieved using &ASSIGN OPT=SETOUT.

SPD

A protected field is created in selector pen detectable format. This enables the terminal operator to select the field using either a LIGHT PEN or the CURSOR SELECT key. SPD field characters must be immediately followed by one of the three designator characters (?, &, or a blank) which can in turn optionally be followed by one or more blanks. A single variable with no other fixed data must also be defined within the field. This single variable must be defined omitting the ampersand (&) and cannot be a system or global variable. If selected by the user, the variable nominated in the SPD field is set to the value SELECTED on return to the NCL procedure. If not selected, the variable is set to a null value.

NULL

An unprotected field is created that allows keyboard entry. However, the field need not contain the name of an input variable to receive data entered in the field as any data entered by the terminal operator in a **TYPE=NULL** field is ignored. Display data in this field can be in any format. The NULL option is supplied to accommodate 4 color terminals where the field attribute byte is used to determine the color in which the field is displayed (7 color terminals utilize an extended datastream to set the color). The NULL option indicates that Panel Services is to use an unprotected field attribute in conjunction with the INTENS operand value, to determine the color of the field.

VALIGN={ <u>NO</u> | LEFT | RIGHT | CENTER | CENTRE }

Applies to output fields only and is ignored if specified for an input field. Determines the alignment of data for an individual variable only. This should not be confused with the JUST operand which applies to field alignment after all variable substitution has been completed. The VALIGN operand is designed to facilitate tabular output without the need to specify many individual field characters on the panel.

The substitution process substitutes the data assigned to a variable in place of the variable name. No additional blanks are created or removed during this process. Thus, if the data being substituted is shorter than the name of the variable itself (for example, the variable &OUTPUTDATA is currently set to 5678) then data following the variable name is shifted left to occupy the area remaining after the removal of the variable name. This destroys any tabular alignment where the length of the data for each variable differs. The VALIGN option ensures that the data to the right of the variable is not shifted to the left if the data being substituted is shorter than the variable name. The length of the variable name (including the ampersand) is the important factor and determines the number of character positions to be preserved during the substitution process.

However, data truncation is not performed and if the data being substituted is longer than the variable name, the data to the right is moved to accommodate all the substituted data. VALIGN works in conjunction with the pad character specified on the PAD operand. The pad character is used to fill any differences between the data being substituted and the length of the variable name being replaced.

VALIGN=NO
: No alignment or padding is performed.

VALIGN=LEFT
: Data is aligned to the left and padded to the right. An abbreviation of L is acceptable.

VALIGN=RIGHT
: Data is aligned to the right and padded to the left. An abbreviation of R is acceptable.

VALIGN=CENTER
: Data is centered (or one position to the left for an odd number of characters) and padded both to the left and to the right. An abbreviation of C is acceptable.

Examples:

```
#FLD    #  TYPE=INPUT REQ=YES EDIT=NUM COLOR=RED RANGE=(1,3)
#FLD    #  BLANKS=TRAIL PAD=_
#FLD    #  TYPE=OUTPUT COLOR=&COLOR HLIGHT=&HLIGHT
#FLD    (  TYPE=INPUT INTENS=HIGH EDIT=DATE4
#FLD    @  HLIGHT=BLINK
#FLD    /  TYPE=SPD
#FLD    %  JUST=R PAD=- -* supplementing default output char
#FLD    _  JUST=ASIS     -* supplementing default input char
#FLD + VALIGN=RIGHT JUST=CENTER
                     -* null pad assumed
```

Notes:

Multiple #FLD statements can be used for the same field character if insufficient space is available on a single statement.

Symbolic variables can be included in a #FLD statement. Variable substitution is performed prior to processing the statement, using variables available to the NCL procedure at the time the &PANEL statement is issued.

The default field characters can be altered using the DEFAULT operand of the #OPT control statement.

The #ERR control statement can be used to greatly simplify the redisplay of a panel to indicate a field in error.

The &CONTROL PANELRC operand can be used to specify that the NCL procedure receives control for further processing when internal validation detects an error in data entered by the operator. When this technique is used the procedure can determine the field in error (from the &SYSFLD variable) and the error message to be issued (from the &SYSMSG variable) and alter its processing accordingly, including altering the text of the error message in the &SYSMSG variable if required.

See Also:

The #OPT, #ERR, and #NOTE panel control statements.

#NOTE

Function:

Allows user comments in a panel definition.

#NOTE	*any text*

Use:

The #NOTE statement provides a means of placing documentation within a panel definition. The #NOTE statement is not processed and is ignored. Multiple #NOTE statements can be specified. However, as with #FLD, #ERR, and #OPT statements, all #NOTE statements must precede the start of the panel, which is determined by the first line that is not a control statement or #NOTE statement.

Operands:

any text
Any free form user text.

Examples:

```
#NOTE This panel is used by the Network Error Log System
#NOTE INWAIT=60 CURSOR=&CURSORFLD
```

Notes:

As shown in the example above the #NOTE statement can provide a simple means of temporarily nullifying another control statement allowing for easy re-instatement when required.

See Also:

The #FLD, #ERR, and #OPT panel control statements.

#OPT

Function:

Defines panel processing options.

```
#OPT    [  ALARM={ YES |_NO_ } ]
        [  BCAST={ YES |_NO_ } ]
        [  CURSOR={ varname | row,column } ]
        [  DEFAULT={ hlu  | X'xxxxxx'  } ]
        [  ERRFLD=varname  ]
        [  FMTINPUT={ _YES_ | NO } ]
        [  IPANULL={ _YES_ | NO } ]
        [  INWAIT=ss.th ]
        [  PREPARSE={ (c,S) | (c,D) } ]
        [  UNLOCK={ _YES_ | NO } ]
        [  MAXWIDTH={ YES | _NO_ } ]
```

Use:

The #OPT statement is a panel control statement used to tailor the processing options for a panel.

Before parsing, the #OPT statement is scanned and any variables are substituted. This makes it possible to dynamically tailor any of the operands on the statement.

Multiple #OPT statements can be specified. However, as with #FLD, #ERR and #NOTE statements, all #OPT statements must precede the start of the panel, which is determined by the first line that is not a control statement.

Operands:

ALARM={ YES | _NO_ }

Determines whether the terminal alarm is to be rung when the panel is displayed. Dynamic control of the alarm can be achieved by changing the value of the ALARM operand using a variable set prior to issuing the &PANEL statement.

If internal validation has detected an error, and the panel is being redisplayed to indicate the error, this operand is ignored and the terminal alarm rung. The #ERR statement can be used to alter the processing performed when an error condition is detected.

BCAST={ YES | _NO_ }

Specifies that the panel is to be redisplayed automatically if a broadcast is scheduled. By default, the only panels that are redisplayed automatically are those that contain one or more of the special broadcast variables, &BROLINE*n*. If BCAST=YES is coded, a broadcast causes the panel to be redisplayed even if it does not contain any of the &BROLINE*n* variables.

CURSOR={ *varname* | *row,column* }

Specifies the name of a variable in either an INPUT or SPD field where the cursor is to be positioned. Alternatively, the precise co-ordinates for the cursor can be defined as *row,column*.

The value of *varname* should be the variable name WITHOUT the ampersand, just as used in the INPUT or SPD field (for example, CURSOR=FIELD5).

Where co-ordinates are specified, *row* must be specified in the range 1 to 62 and *column* in the range 1 to 80. The *row* and *column* values are always relative to the start of the current window and therefore remain unchanged when operating in split screen mode. The &CURSROW and &CURSCOL system variables can be used to determine the location of the cursor on input to the system.

Dynamic positioning of the cursor can be achieved by using a variable or variables (including the ampersand) in place of *varname* or *row,column*. The invoking NCL procedure can set the variables to the name of the field to contain the cursor or the coordinates prior to issuing the &PANEL statement.

If internal validation detects an error, and the panel is being redisplayed to indicate the error, the CURSOR operand is ignored and the cursor is positioned to the field in error.

Specifying *varname* with a name other than the name of a variable used in an INPUT or SPD field results in an error. If coordinates are used, and they lie outside the dimensions of the window currently displayed, the cursor is positioned in the upper lefthand corner of the window.

DEFAULT={ *hlu* | X'*xxxxxx*' }

This operand alters the three standard default field characters. If the #OPT statement is omitted, or the DEFAULT operand not used, three standard field characters are provided for use when defining the panel. They are:

% = protected, high-intensity

\+ = protected, low-intensity

_ = unprotected, high-intensity

It might be necessary to select alternative field characters, for example if the underline character is required within the body of the panel for some reason.

The DEFAULT=*hlu* operand must always specify three characters. The characters chosen must be non-alpha and non-numeric, that is, any special character except ampersand (&), which is reserved for variables. They must not duplicate another field character, except one already defined as a default.

The order of the characters is significant, as the attributes of the standard default characters apply in the order described above.

Therefore specification of DEFAULT=*+/ results in:

*	=	protected, high-intensity
+	=	protected, low-intensity
/	=	unprotected, high-intensity

You can also specify the default field characters in hexadecimal in the format:

DEFAULT=X'*xxxxxx*'

where each *xx* pair represents a hexadecimal number in the range X'00' to X'FF'. All numbers except X'00' (null), and X'40' (blank) or X'50' (ampersand), are valid. This even allows alphanumeric characters to be used as field characters. For example, if you specify X'C1', any occurrence of the letter A in the panel definition is treated as the default character. (It is advisable to use hexadecimal values that do not correspond to alphanumeric characters.)

For example, specification of DEFAULT='010203' results in:

X'01'	=	protected, high-intensity
X'02'	=	protected, low-intensity
X'03'	=	unprotected, high-intensity

ERRFLD=*varname*

Specifies the name of a variable in an INPUT field which is in error and for which error processing is to be invoked as defined on a #ERR statement. Use of this operand without including a #ERR statement within the panel definition results in an error. The ERRFLD operand provides a simple way of informing Panel Services that the field identified by *varname* is in error. Panel Services displays the panel using the options defined on the #ERR statement. The #ERR statement could indicate that the error field is to be displayed in reverse-video, colored red and the terminal alarm sounded. Use of the ERRFLD operand can be accompanied by the assignment of some error text into a variable appearing on the screen that identifies the nature of the error.

The operand can be specified with the name of a variable (including the &) that is set to null unless an error occurs, in which case the NCL procedure sets the variable to the name of the field in error prior to issuing the &PANEL statement to display the panel.

ERRFLD provides the panel designer with a simple means of changing the attributes of a field (such as color and high-lighting) without the need to resort to dynamic substitution of #FLD statements.

Consider the case where an input field &INPUT1 is found to be in error and the #OPT statement has been defined with ERRFLD=&INERROR. The NCL procedure assigns the name of the variable used to identify the input field, in this case INPUT1 (minus the &), into &INERROR, and then redisplays the panel.

```
&INERROR = INPUT1
&SYSMSG  = &STR THIS FIELD IS WRONG
&PANEL  MYPANEL
```

Note

In this example the text that identifies the nature of the error has been assigned into the variable &SYSMSG which is defined somewhere on the panel.

The same effect can be achieved by using the &ASSIGN OPT=SETERR verb. This allows more than one field to be marked as in error.

Note

ASSIGN OPT=SETERR is only effective if &CONTROL FLDCTL is in effect.

FMTINPUT={ YES | NO }

Determines if input fields are to be formatted when a panel is displayed. This is a specialized option that is designed to be used in conjunction with INWAIT. When processing with INWAIT, the time interval might expire at the instant when data is being entered by the operator. If the same panel is redisplayed to update the screen contents, the data entered by the operator is lost as the new panel is written. FMTINPUT can be used to bypass formatting of input fields and hence when the panel is redisplayed only output fields are written. The value of this operand can be assigned to a variable from within the NCL procedure and changed between YES and NO as required (#OPT FMTINPUT =&YESNO). Care must taken when using this facility, as incorrect use of FMTINPUT=NO can result in validation errors. Ideally, a panel should be displayed initially with FMTINPUT=YES and only when the INWAIT timer expires is it redisplayed with FMTINPUT=NO.

IPANULL={ YES | NO)

The default YES, specifies that if the panel is displayed with the INWAIT option and the time specified on the INWAIT expires so that control is returned to the procedure without any panel input, or input is caused by a PA key, all variables associated with the panel input fields are to be set to null value.

If you do not want input field variables to be erased, if INWAIT completes, or a PA key is pressed, specify IPANULL=NO.

INWAIT=*ss.th*

Specifies the time in seconds and/or parts of seconds that Panel Services is to wait for input from the terminal prior to returning control to the NCL procedure following the &PANEL statement. By default the system waits indefinitely for input having displayed a panel. This might not always be desirable, as is the case where a terminal is performing a monitoring function where input might be infrequent or never occur. If INWAIT is utilized and the specified time elapses, control is returned to the NCL procedure with all input or SPD variables set to null. Should input be made during the time interval, the time period is cancelled and standard processing will proceed.

The maximum value that can be specified for INWAIT is 86400.00 seconds (24 hours).

Specification of part seconds is possible. For example:

INWAIT=.5
INWAIT=20.5
INWAIT=.75

Any redisplay of a panel, for example, by using the clear key, causes the panel to be redisplayed and the time interval reset.

Specification of internal validation options, such as REQUIRED=YES are ignored if the time interval expires before input is received.

Specification of INWAIT=0 or INWAIT=0.00 indicates that no input is to be accepted and control is returned to the NCL procedure immediately after the panel has been displayed. In this case the period that the panel remains displayed is determined by subsequent action taken by the procedure.

The invoking NCL procedure can determine if the INWAIT time elapsed or if data was entered by testing the &INKEY system variable. &INKEY is set to the character value of the key pressed by the operator to enter the data (for example, ENTER or F1). If the INWAIT time interval elapsed, and no entry was made, the &INKEY variable is set to null. If processing with &CONTROL PANELRC is in effect, &RETCODE is set to 12 to indicate that the INWAIT timer has expired.

Note

INWAIT is ignored for asynchronous panels.

PREPARSE={ (c,S) | (c,D) }

Preparsing provides a means for dynamically modifying the location of field characters in a panel. The position of field characters (as defined by the #FLD control statement) is determined when the panel is created by Panel Services and remains fixed until the panel is modified.

Although the attributes of each field character (such as the color of the field) can be modified by the use of variables in the #FLD statement, this technique is limited in the number of variations that can be achieved.

The PREPARSE operand requests that Panel Services perform a preliminary substitution scan of each panel line prior to processing the line for field characters. The PREPARSE operand specifies a substitution character *c* that is to be used to determine where substitution is to take place. This character is processed in exactly the same manner as an ampersand (&) is processed during standard substitution.

The ability to specify a character other than an ampersand means that preparsing does not impact standard substitution when it is performed following preparsing. Preparsing can be used to alter a field character that appears in a particular position, thereby allocating a new set of attributes to the field or to create entire new fields (or complete lines) that in themselves contain the required field characters.

(*c*,S)

Indicates that the character *c* is to be used as the preparse character for the panel, but that the Static Preparse Option is to apply during preparse processing. This prevents the movement of preparse or field characters during the substitution process. This option is useful when panels are being dynamically modified to hold data which can vary in length but is to be displayed in columns. If necessary, substituted data can be truncated if it is too long to fit into its target field without overwriting the next occurrence of a preparse or field character on the same line.

(*c*,D)

Indicates that the character *c* is to be used as the preparse character for the panel, but that the Dynamic Preparse Option is to apply during preparse processing. The dynamic option allows the movement of preparse or field characters to left or right of their original position to accommodate differing lengths of data being substituted into the panel.

UNLOCK={ YES | NO }

Determines if the terminal keyboard is to be unlocked when the panel is displayed. Specification of UNLOCK=NO prevents entry of data by the terminal operator, and can be used in conjunction with the INWAIT operand where a panel was being displayed for a short period, prior to progressing to some other function.

MAXWIDTH={ YES | NO }

Specify MAXWIDTH=YES to indicate the display is to be wider than the standard 80 character width. MAXWIDTH=YES means that panel services will use the number of columns available on the terminal (&ZCOLS).

Examples:

```
#OPT DEFAULT=#$%
#OPT INWAIT=60 CURSOR=&CURSORFLD
#OPT CURSOR=IN1 ALARM=YES
#OPT ALARM=&ALARM PREPARSE=($,D)
#OPT ERRFLD=&INERROR
#OPT INWAIT=.5 UNLOCK=NO PREPARSE=($,S)
#OPT CURSOR=5,75
#OPT CURSOR=&ROW,&COLUMN FMTINPUT=&FMT
```

Notes:

Multiple #OPT statements can be used if required.

Symbolic variables can be included in a #OPT statement. Variable substitution is performed prior to processing the statement.

Panel redisplay following use of the CLEAR key is automatic. Control is not returned to the invoking NCL procedure. The attributes of the standard default characters can be modified using a #FLD statement that adds additional attributes (such as color) or alters existing attributes.

See Also:

The #FLD, #ERR, and #NOTE panel control statements.

#TRAILER

Function:

Provides a means of placing specified lines at the end of the screen, regardless of screen size.

```
#TRAILER      [ START | END ]
              [ POSITION={ YES | NO } ]
```

Use:

The #TRAILER statement can be used to position function key prompts at the bottom of the screen.

Indicate the start of the trailer lines with a #TRAILER START statement. Then enter the lines to appear at the bottom of the screen, followed by a #TRAILER END statement.

Operands:

START

Indicates the start of the lines to be placed in the trailer. Each line following this line until a #TRAILER END statement or another control statement such as #FLD is placed in the trailer.

END

Indicates that this is the end of the lines to be placed into the trailer. There must have been a #TRAILER START statement earlier in the panel definition. No other operands can be specified on a #TRAILER END statement.

POSITION={ YES | NO }

Specifies if the trailer lines are to be displayed. The values available are:

YES

The trailer lines are displayed on the final lines of the physical screen.

NO

This value can be used to suppress the display of the trailer lines, even though they remain in the panel definition.

Examples:

```
#TRAILER START
%This appears on the last line of the panel
#TRAILER END
```

Note

1. The #TRAILER statements must appear before the first panel line in the definition. If you want to preparse the lines, you must place the #TRAILER statements after the #OPT PREPARSE= statement.

2. The field attribute characters which you use in the trailer lines can be defined before or after the trailer lines in the panel.

3. The trailer lines cover any panel lines that would otherwise have been displayed.

4. The trailer lines are positioned so that they end at the bottom of the physical screen if the window starts at the top of the screen.

7

Maintaining Menus

This chapter describes the Common Application Services (CAS) Menu facility.

CAS builds and presents menus, manages user interaction with menus, and provides you with facilities for maintaining menu definitions.

About Menu Definitions

A menu definition contains the information required to build a menu and display it to the user.

A menu definition consists of the following:

- A *menu description* that includes the menu identifier, its title and other presentation settings
- A series of *menu options* to be displayed on the menu and their corresponding actions
- A series of *menu input fields* when data needs to be entered on a menu panel in support of a particular option

Defining a Menu

You can create a menu by selecting the Add Menu option from the CAS : Menu Definition Menu (or by choosing the ADD action from a list of menu definitions) or by copying an existing menu definition. For details of how to copy a menu definition see *Copying a Menu Definition*, on page 7-11.

The Menu Definition Maintenance Menu

Select option **M** from the CAS : Maintenance Menu to display the CAS : Menu Definition Menu, shown in Figure 7-1.

Figure 7-1. CAS : Menu Definition Menu

```
SOLVPROD----------------- CAS : Menu Definition Menu --------------------$MH010
Select Option ===>

   A   - Add Menu
   B   - Browse Menu
   U   - Update Menu
   D   - Delete Menu
   C   - Copy Menu
   L   - List Menus
   V   - View Menu
   X   - Exit

Appl ID ...........+ ___ ( Required B U D C V Optional A L )
Menu Number ........ ___ ( Required B U D C V Optional A )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the CAS : Menu Definition Menu are as follows:

Appl ID
The 3 character identifier of the application to which the menu belongs

Menu Number
The 3 digit identifier of the menu; must be a number between 1 and 225.

The following options are available on the CAS : Menu Definition Menu:

Option	Explanation
A	Add a new menu definition
B	Browse an existing menu definition
U	Update an existing menu definition
D	Delete a menu definition
C	Copy an existing menu definition
L	List menu definitions
V	View a menu in display format

These menu options are described below.

Adding a Menu Definition

Defining a menu requires you to fill in three panels. The first of these contains menu identification and presentation details; the second defines the menu options and their corresponding actions; and the third specifies the input field(s) required for each menu option.

The first panel is the CAS : Menu Description panel, as shown in Figure 7-2.

Figure 7-2. CAS : Menu Description Panel

```
SOLVPROD------------------- CAS : Menu Description -----------------Page 1 of 3
Command ===>                                                    Function=Update

Appl ID ...................+ TST
Menu Number ................ 001

Menu Title ................. Test Application Menu_____________________________

Is This a Primary Menu? .... YES (YES or NO)
Display Userid Info Box? ... NO_ (YES or NO)

Menu Service Procedure ..... ________

Top Left Corner Display .... SOLVPROD____
Top Right Corner Display ... TST001______

Menu Input Field Attributes:
Mandatory Input ............ _
Optional Input ............. _
High Intenstity Output ..... _
Low Intensity Output ....... _

 F1=Help      F2=Split     F3=File      F4=Save
                           F9=Swap                 F11=Page 2   F12=Cancel
```

The fields displayed on the CAS : Menu Description panel are as follows:

Appl ID

The 3-character identifier of the application to which this menu definition belongs.

Menu Number

The 3-digit identifier of the menu; must be a number between 1 and 225.

Menu Title

The (up to) 50-character title of the menu. When the menu is displayed, the text in this field is centered on the title line.

Is This a Primary Menu?

This field determines whether this menu is treated as a primary menu:

- Enter **YES** to specify a primary menu—that is, issuing a RETURN command from a lower level function causes this menu to be redisplayed.
- Enter **NO** to specify that this is not a primary menu—that is, issuing a RETURN command from a lower level function causes this menu to be skipped and the previous primary menu to be displayed.

Display Userid Info Box?

Determines whether the *Userid Info Box* is displayed on the menu (enter **YES**) or not (enter **NO**). The *Userid Info Box* contains the user's identifier and VTAM LU, plus the current time and date.

This field is mandatory.

Menu Service Procedure

Allows you to (optionally) specify the name of a procedure to act as the service procedure for the menu.

Top Left Corner Display

Allows you to (optionally) specify text or a variable to be displayed in the left corner of the title line. Typically used to display the system identifier.

Top Right Corner Display

Allows you to (optionally) specify text or a variable to be displayed in the right corner of the title line. Typically used to display the menu identifier.

Menu Input Field Attributes:

These fields allow you to specify which characters are to be used as presentation attributes when defining input fields on the CAS : Menu Input Fields panel (see Figure 7-4).

Mandatory Input

Allows you to (optionally) specify the character to be used as the field attribute of Mandatory input. The default value for this field is the underscore (_) character.

Optional Input

Allows you to (optionally) specify the character to be used as the field attribute of Optional input. The default value for this field is the backslash (\ ) character.

High Intensity Output

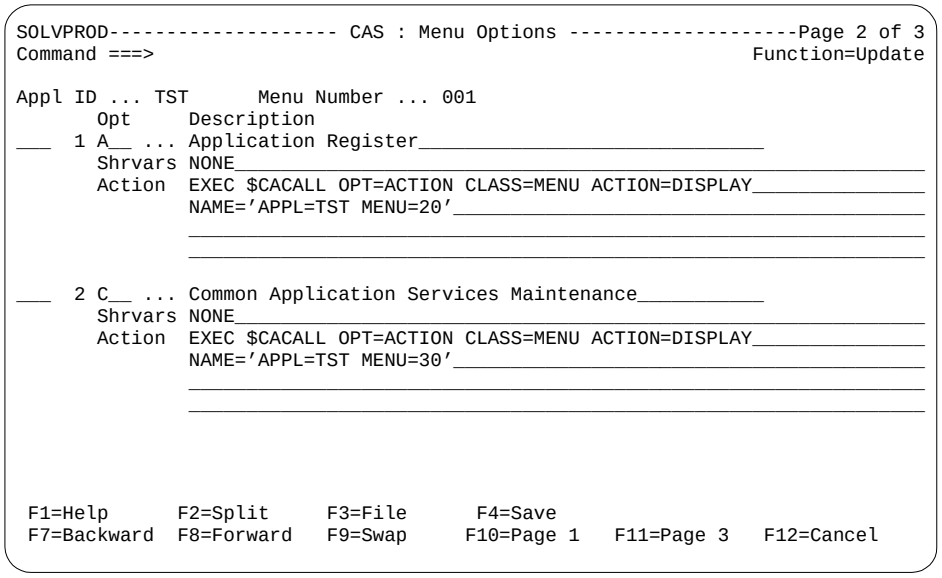
Allows you to (optionally) specify the character to be used as the output field attribute of High. The default value for this field is the separator (¡) character.

Low Intensity Output

Allows you to (optionally) specify the character to be used as the output field attribute of Low. The default value for this field is the single quote (¡) character.

After completing the menu definition details, press the PAGE 2 key to go to the CAS : Menu Options panel (as illustrated in Figure 7-3). Specify menu options using this panel.

Figure 7-3. CAS : Menu Options Panel



Enter the following four fields once for each option to be displayed on the menu. (You can define up to 15 options; use the FORWARD and BACKWARD keys, to scroll between them.)

Opt

The option mnemonic that is entered by a user to select a menu option. It can be between 1 and 3 alphanumeric characters.

Description

The text that is displayed on the menu to describe the option.

Shrvars

Specifies which variables can be shared between the menu service procedure and the procedure invoked by the **Action**. Valid values for this field are:

ALL

Share all variables (except those with the prefix #MH, which are reserved for internal use by CAS)

NONE

Do not share any variables (this is the default value).

pref,pref

Share variables with these prefixes.

pref,pref

Do not share variables with these prefixes.

Note

The **SHRVARS** field is only meaningful if the **Action** for the menu option contains an EXEC command.

Action

Specifies the action that corresponds to this option; can be up to 4 lines in length.

An **Action** generally consists of an EXEC command, in the following format:

- The word **EXEC**
  - The name of the NCL procedure to be executed
  - Any parameters required by the procedure

For instance, the **Action** on the second menu option of the sample panel (Figure 7-3) is:

```
EXEC $CACALL OPT=ACTION ACTION=DISPLAY CLASS=MENU
NAME='APPL=TST MENU=30'
```

This **Action** displays another menu by invoking $CACALL—see Chapter 18, *CAS Programming Interface ($CACALL)* for details.

Warning

&CONTROL RESCAN is in effect when an **Action** procedure is invoked. To counteract this, issue &CONTROL NORESCAN on entry (see the *Network Control Language User's Guide* for details).

To assist you in defining menu options, the following line editor commands are available (see Appendix B, *Text Editor Commands,* for full details):

Command	Action
A	Move/copy after this line
B	Move/copy before this line
C	Copy line(s)
D	Delete line(s)
M	Move line(s)
R	Repeat line(s)

If you do not want any input fields to appear on the menu, then the definition is complete when you have finished specifying the menu options. Press the FILE key to save the menu definition.

If any of the options on your menu require data from the user, you need to define input fields. Press the PAGE 3 key to go to the CAS : Menu Input Fields panel, as illustrated in Figure 7-4.

Figure 7-4. CAS : Menu Input Fields Panel

```
SOLVPROD------------------ CAS : Menu Input Fields -----------------Page 3 of 3
Command ===>                                                    Function=Update

Appl ID ... TST      Menu Number ... 001
Use the attributes below to build the field input line beneath Related Options.
  _=Input (Mandatory)   \=Input (Optional)   ¡=Output (High)   ‘=Output (Low)
___  1  Required for ... A_____________________________________________________
        Optional for ... _____________________________________________________
‘Application ID.....\#LH‘     (Required !A‘)_____________________________________

___  2  Required for ... _____________________________________________________
        Optional for ... _____________________________________________________
_____________________________________________________________________________

___  3  Required for ... _____________________________________________________
        Optional for ... _____________________________________________________
_____________________________________________________________________________

___  4  Required for ... _____________________________________________________
        Optional for ... _____________________________________________________
_____________________________________________________________________________

 F1=Help      F2=Split     F3=File      F4=Save
 F7=Backward  F8=Forward   F9=Swap     F10=Page 2                F12=Cancel
```

Enter the following data on the CAS : Menu Input Fields panel to define up to 15 input fields on the menu (use the FORWARD and BACKWARD keys to scroll between them).

Required for

Specifies any menu options for which this input field is mandatory. Enter the appropriate menu option mnemonics separated by spaces.

Optional for

Specifies any menu options for which this input field is optional. Enter the appropriate menu option mnemonics separated by spaces.

input field

This is where you specify the actual input field as it appears on the menu. You need to include the name of a variable to receive the input data.

The line editor commands available in the CAS : Menu Options panel (as described in Appendix B, *Text Editor Commands*) are available on this panel.

After specifying the menu's input fields, press the FILE key to add the menu definition. To cancel the menu specification, press the CANCEL key.

Maintaining Menu Definitions

This section describes the menu options that are used to maintain existing menu definitions.

You can specify the menu that you want to update by using the CAS : Menu Definition Menu. On this menu you must enter the **Appl ID** and **Menu Number** fields and select the update option, or you can select a menu definition from a list.

Listing Menu Definitions

Option **L** displays the CAS : Menu Definition List over two screens as shown in Figure 7-5 and Figure 7-6. Use the RIGHT and LEFT keys to move between the screens.

Figure 7-5. CAS : Menu Definition List Panel (screen 1)

```
SOLVPROD----------------- CAS : Menu Definition List --------------------------
Command ===>                                                 Scroll ===> PAGE

                                  S/B=Browse U=Update D=Delete C=Copy V=View
    Menu ID    Menu Title                                              File ID
    ZAM010     Asset : Primary Menu                                    MODSDIS
    ZAM020     Asset : Assets Menu                                     MODSTST
    ZAM030     Asset : Expenses Menu                                   MODSTST
    ZAM040     Asset : Services Menu                                   MODSTST
    ZAM050     Asset : Agreements Menu                                 MODSTST
    ZAM080     Asset : Primary Menu (Administrator)                    MODSDIS
    ZAM090     Asset : Service Level Reporting                         MODSTST
    ZCG010     Change : Primary Menu                                   MODSTST
    ZCG020     Change : Lists Menu                                     MODSTST
    ZCG030     Change : Primary Menu (Administrator)                   MODSTST
    ZCO010     Configuration : Primary Menu                            MODSDIS
    ZCO020     Configuration : Environmental Planning Menu             MODSTST
    ZCO030     Configuration : Build/Verify Menu                       MODSUSR
    ZCO031     Configuration : VPD Interface Menu                      MODSTST
    ZCO032     Configuration : LAN Interface Menu                      MODSTST
    ZCO033     Configuration : Staging File Maintenance Menu           MODSTST
    ZCO040     Configuration : Services Menu                           MODSTST
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 7-6. CAS : Menu Definition List Panel (screen 2)

```
SOLVPROD----------------- CAS : Menu Definition List --------------------------
Command ===>                                                 Scroll ===> PAGE

                                  S/B=Browse U=Update D=Delete C=Copy V=View
    Menu ID    Created      Last Updated
    ZAM010     17-NOV-1992  17-NOV-1992 00.00 INSTALL
    ZAM020     17-NOV-1992  08-DEC-1992 08.59 USER01
    ZAM030     17-NOV-1992  07-DEC-1992 17.36 USER01
    ZAM040     17-NOV-1992  07-DEC-1992 17.36 USER01
    ZAM050     17-NOV-1992  07-DEC-1992 17.37 USER01
    ZAM080     17-NOV-1992  17-NOV-1992 00.00 INSTALL
    ZAM090     25-FEB-1992  09-DEC-1992 14.26 USER01
    ZCG010     17-NOV-1992  07-DEC-1992 17.37 USER01
    ZCG020     17-NOV-1992  07-DEC-1992 17.38 USER01
    ZCG030     17-NOV-1992  07-DEC-1992 17.38 USER01
    ZCO010     17-NOV-1992  17-NOV-1992 00.00 INSTALL
    ZCO020     17-NOV-1992  07-DEC-1992 17.38 USER01
    ZCO030     09-APR-1992  16-DEC-1992 10.38 USER01
    ZCO031     09-APR-1992  18-AUG-1992 15.55 USER01
    ZCO032     09-APR-1992  18-AUG-1992 15.55 USER01
    ZCO033     09-APR-1992  13-AUG-1992 15.22 USER01
    ZCO040     17-NOV-1992  07-DEC-1992 17.39 USER01
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left
```

If you enter an application identifier in the **Appl ID** field when you select this menu option, the list will only contain menu definitions that begin with that application identifier.

The fields displayed on the Menu Definition List are as follows:

Menu ID
The menu identifier.

Menu Title
The title of the menu.

File ID
The identifier of the MODS control file in which the definition is stored. This is the highest CAS concatenation level at which the menu is present.

Created
The date the menu definition was created.

Last Updated
The date and time the menu definition was last updated and the identifier of the user who updated the definition.

You can perform the following actions on any menu in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B, S, or /	Browse the selected menu definition
U	Update the selected menu definition
D	Delete the selected menu definition
C	Copy the selected menu definition
V	View the menu in display format

These actions are described below. These actions can also be performed by using the equivalent menu options.

Browsing a Menu Definition

Option **B** displays a menu definition, as shown in Figure 7-2, Figure 7-3, and Figure 7-4. Use the PAGE 2 and PAGE 3 keys to move between panels.

Updating a Menu Definition

Option **U** displays the Menu Description panel (as illustrated in Figure 7-2), the Menu Options panel (Figure 7-3) and the Menu Input Fields panel (Figure 7-4) except that the function is Update and the menu identifier cannot be modified.

Modify the fields on the panel as required; refer to the section titled *Adding a Menu Definition*, on page 7-3 for details.

Deleting a Menu Definition

Option **D** displays the CAS : Menu Description panel with a message requiring you to confirm the deletion. Press the ENTER key to delete the menu definition, or press the CANCEL key to cancel the deletion.

Copying a Menu Definition

To copy an existing menu definition to another (new) menu definition, select option **C** from the CAS : Menu Definition Menu. You must complete the **Application ID** and **Menu Number** fields to identify the menu definition to copy from.

This option displays the Menu Description panel (as illustrated in Figure 7-2), the CAS : Menu Options panel (Figure 7-3) and the CAS : Menu Input Fields panel (Figure 7-4) of the new menu. Modify the fields on these panels as required; refer to the section titled *Adding a Menu Definition*, on page 7-3 for field details.

After completing the new menu definition, press the FILE key to save it. To cancel the copy, press the CANCEL key.

Viewing a Menu Definition

Option **V** displays the menu as the user will see it, as shown in Figure 7-7. This is the menu defined in Figure 7-2, Figure 7-3 and Figure 7-4.

Figure 7-7. Viewing a Menu That You Have Defined in Display Format.

```
SOLVPROD------------------- Test Application Menu -----------------------TST001
Select Option ===>

   A   - Application Register
   C   - Common Application Services Maintenance
   X   - Exit

Application ID..... ___      (Required  A )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

8

Maintaining Lists

This chapter describes the Common Application Services (CAS) List facility.

Lists are defined and then stored on a database. They can be recalled to display a series of items as required.

This chapter describes how to create and maintain list definitions.

About Lists

A list displays a series of items on a panel. These items are typically objects and are identified with a series of attributes that relate to the object. For example, they may be identified with the object's identifier and description.

The list facility provides a generalized method for displaying and allowing selection of list items. The same list definition can be used to display four different types of lists:

- **Action List**
  Displays a series of objects against which actions can be applied
- **Single Select**
  Displays a series of items from which one is selected
- **Multiple Select**
  Displays a series of items from which one or more are selected
- **Numbered List**
  Displays a series of items which are numbered—an item is selected by entering its corresponding number

The type of list that is created from a list definition is determined by the call to to the CAS interface, $CACALL. For details see Chapter 18, *CAS Programming Interface ($CACALL)*.

To create a list you must specify a list definition. A list definition comprises the following:

- Identifying information about the list and a specification of its behavior
- The source of the data to be displayed on the list
- A service procedure which is used to retrieve items to be included in the list
- A criteria definition to be used to determine an item's eligibility for inclusion on the list
- A sort expression to determine how items in the list are sorted
- Line entry presentation attributes which allow you to highlight specific records in a list based on a condition that you specify
- The format of the list

A list is defined using five panels that specify these details:

- List Description panel
- List Criteria panel
- List Entry Line Presentation Attributes panel
- List Format panel
- List Entry Line Fields panel

Each of these are described briefly below.

List Description Panel

The list description panel is used to specify identifying information about the list and its behavior.

You can specify whether a list is active or inactive. Inactive list definitions cannot be used and do not appear on a list of lists. A list can also be defined as private to a specific user—unlike a public list which is available to all users.

Lists can, optionally, be defined as belonging to a *group*. A group is a logical collection of lists and represents a convenient way of displaying only those lists that are relevant when a list of lists is displayed. For example, you could assign all lists relating to problem management to a group.

The list service procedure retrieves list items and processes requests to perform actions against list items. A list service procedure is dependent on the list's data source.

You can specify the identifier of the help definition to be used when help is requested in the list panel. In addition you can specify the name of a list exit procedure to perform application specific processing.

List Criteria Panel

This panel is used to optionally specify a criteria, sort expression and existing list format to be used by the list definition.

A criteria can be associated with a list (that is used by the service procedure) to filter items to be included on the list. This criteria can be an existing criteria definition or a freeform criteria in which the user enters the details of the criteria at run time.

You can specify a sort expression which is used to determine the order that items are displayed. The sort expression is used by the list service procedure and thus the syntax and complexity of this expression is dependent on this procedure. If a sort expression is not specified, items are displayed in the order in which they are retrieved by the service procedure.

This panel also allows you to specify the identifier of an existing list definition from which the format and presentation attributes for the current list are drawn.

For example, if you want to define several lists that share the same format, you can define a list to be used as a template. This list should be set to INACTIVE so that it cannot be used, and does not appear on a list of lists. You can then define subsequent lists in the same format as the template list.

List Entry Line Presentation Attributes

This panel allows you to specify presentation attributes (color and highlighting) for specific records based on a condition (a boolean expression) that you specify.

This means that you can highlight particular records of interest in a list to make them easier to identify.

For example, consider a list of problem records. You can specify a condition to identify all problem records that have the status of OPEN and set the display attributes to show those records in, for example, red, flashing text.

List Format Panel

A list's format determines how list items are displayed, the other attributes that are displayed about each item, and static heading and option text. This panel displays a text editor that is used to define the format of a list.

A list format consists of:

- Up to 4 comment lines; one for each list type
- Up to 10 screen formats. Each screen format defines a screen that is displayed when the list is presented to a user (that is, a list can be presented over ten screens). The RIGHT and LEFT keys can be used to scroll between the screens.

  A screen format consists of:

  - Up to 10 heading lines
  - An entry line, which contains the list data. (Each line of a displayed list is referred to as an entry line.)

List Entry Line Fields Panel

This panel is used to assign aliases to attribute identifiers used on the list format panel.

Entry line fields can sometimes be longer than the data to be displayed, meaning the NCL variable to be defined in the entry line might not fit.

To overcome this, you can define a shorter variable name in the entry line, and then assign the name of the real field that is to be substituted when you display a list using the definition.

The value of an entry line field can be the same as that of the associated real field (when its length does not cause formatting difficulties).

List Cache

When a list definition is recalled for use it is loaded into a VARTABLE for optimum performance. This is referred to as the List Cache. As different lists are used they are added to this cache.

Defining a List

You can define a new list by using the Add List option from the List Definition Menu (or the ADD action from the List Definition List) or by copying an existing list. See *Copying a List Definition*, on page 8-20.

List Definition Menu

Lists are accessed through the CAS : List Definition Menu (option **L** of the CAS : Maintenance Menu).

This panel allows you to add, browse, update, delete and copy lists, and to obtain a list of lists and also allows you to reset the list cache. The CAS : List Definition Menu is illustrated in Figure 8-1.

```
SOLVPROD------------------ CAS : List Definition Menu --------------------$LH010
 Select Option ===>

    A   - Add List
    B   - Browse List
    U   - Update List
    D   - Delete List
    C   - Copy List
    L   - List Lists
    R   - Reset List Cache
    X   - Exit

 Appl ID ....+ ___           ( Required B U D C  Optional A L )
 List Type ... _______       ( Required B U D C  Optional A L )
 Userid ..... ________       ( Optional A B U D C L )
 List Name ... ________      ( Required B U D C  Optional A L )
 Group ......+ ____________ ( Optional L )




  F1=Help      F2=Split     F3=Exit      F4=Return
                            F9=Swap
```

Figure 8-1. CAS : List Definition Menu

The input fields displayed on the CAS : List Definition Menu are as follows:

Appl ID
The 3 character identifier of the application to which the list belongs.

List Type
A list is either:

- **PUBLIC** (available to all users)
- **PRIVATE** (available only to the owner of the list)

Userid
If the **List Type** is PRIVATE, the user ID of the owner must be entered (8 characters).

List Name
The name of a list is unique within an application and type. The name must be from 1 to 8 alphanumeric or national characters, or both.

Group
The name of a logical grouping of lists. When selecting option **L**, you can enter a **Group** name in this field to specify a group of lists to be listed.

The following options are available on the List Definition Menu:

Option	Explanation
A	Add a new list definition
B	Browse an existing list definition
U	Update an existing list definition
D	Delete a list definition
C	Copy an existing list definition
L	List list definitions
R	Reset the list cache

Adding a List Definition

Select option **A** from the CAS : List Definition Menu to add a new list definition. This displays the CAS : List Description panel, as shown in Figure 8-2.

You can define a new list by using the Add List option from the List Definition Menu (or the ADD action from the List Definition List) or by copying an existing list. See *Copying a List Definition* in this chapter.

```
SOLVPROD------------------ CAS : List Description -----------------Page 1 of 4
Command ===>                                                      Function=Add

Appl ID ..............+ ___
List Type ............. ________   (PUBLIC or PRIVATE)
Userid ................ _________  (Userid if PRIVATE)
List Name ............. _________
Description ........... _____________________________________
Title ................. ____________________________________________________
Status ................ ACTIVE__        Group ..............+ _____________
Service Procedure ..... _________       Data Source ......... _____________
Get All Entries? ...... YES             Exit Name ........... _________
Add Allowed? .......... YES             Help Name ........... _____________
Default Mnemonic ...... B__             Select Mnemonic ..... S__
Entry Msg Position .... 2__             Entry Msg Length .... ___
Present Empty List? ... YES             Auto Refresh Rate ... _____
                                        Heading Sub Char .... ___
Comments .............. ____________________________________________________
                        ____________________________________________________
                        ____________________________________________________
                        ____________________________________________________

 F1=Help      F2=Split     F3=File      F4=Save
              F8=Forward   F9=Swap                              F12=Cancel
```

Figure 8-2. CAS : List Description Panel

The input fields displayed on the CAS : List Definition panel are as follows:

Appl ID

The 3 character identifier of the application to which the list belongs.

List Type

A **List Type** is either:

- **PUBLIC** (available to all users)
- **PRIVATE** (available only to the owner of the list)

Userid

If a **List Type** is PRIVATE, the user ID of the owner must be entered (8 characters).

List Name

The name of a list is unique within an application and type. The name must be from 1 to 8 alphanumeric or national characters, or both.

Description

A 1 to 32 character description of the list. The description is displayed when lists are listed for selection purposes, and should thus be as informative as possible.

Title

The text of the title to be displayed when a list is presented; it can be up to 50 characters in length. The title can include constant data, variable data or both. Constant data is static and variable data is substituted into the title when the list is displayed. Variable data is specified as NCL variables (for example, the NCL variable &USERID in a title is replaced by the user's user ID when the list is constructed). Variable data to be included in the title can be set by the list exit procedure or service procedure.

Inclusion of a title in a list definition is optional. If the **Title** field is left blank, the **Description** field is displayed as the title when the list is presented.

Status

The status of a list can be either:

- **INACTIVE**, meaning that the list cannot be displayed and is not included in a list of lists.
- **ACTIVE**, meaning the list can be displayed and is included in a list of lists.

Group

The name of a logical grouping of lists. The group name must be defined in the table $ADGROUP. See 4, *Registering an Application* for information on maintaining application groups.

Service Procedure

Specifies the 1 to 8 character name of an NCL procedure that retrieves the entries to be included in a list. For action lists, this procedure also processes user requests to apply actions to items on the list. See Chapter 20, *List Service Procedure Interface*, for further details about the service procedure.

Data Source

This (optional) field is used by the service procedure to determine the source from which data for the list is to be obtained.

Get All Entries?

This field determines how entries in a list are retrieved for presentation:

- Enter **YES** to retrieve all entries in a list before the list is presented. For small lists (that is, lists that fit on one screen), this type of retrieval gives the best performance.
- Enter **NO** to retrieve a screenfull of entries at a time. For large lists (multiple screens), this type of retrieval gives the best performance.

Exit Name

This (optional) field allows you to specify an NCL procedure to be executed when the list is displayed. The procedure provides any variable data required by the **Title**, the **Sort Expression**, the **Comment Lines,** or the screen formats (see Figure 8-5).

See Chapter 21, *List Exit Procedure Interface*, for further details about the exit procedure.

Add Allowed?

This field determines whether the ADD command is supported by this list, that is whether you can add a new record from the list.

- Enter **YES** to assign the ADD command to a function key
- Enter **NO** to disable the ADD command

Help Name

This (optional) field allows you to specify a function-level help definition to be displayed when the user requests help on the list.

If this field is left blank, the application level help file for the application to which the list belongs is displayed. See Chapter 10, *Maintaining Help*, for details of how to define this help.

Default Mnemonic

The mnemonic assigned to the action that is applied when an entry is selected from an Action List using either the select mnemonic, the forward slash character (/), or cursor selection.

Select Mnemonic

This (optional) field specifies the mnemonic that can be entered by a user to select entries on a Single Select List or Multiple Select List. If the select mnemonic is entered on an Action List, the action assigned to the default mnemonic is applied to the selected entry.

On Action Lists, Single Select Lists, and Multiple Select Lists, the forward slash character (/) is accepted as valid and treated as the select mnemonic.

The select mnemonic and the slash character are not used by a Numbered List.

Entry Message Position

A number between 2 and 75. Specifies the position on the list entry line where the entry message begins.

An entry message is a message that is displayed on an entry line, overlaying the line's data. The entry message is set by the list's service procedure. For example, an entry message could be set to `**DELETED**` when an entry is deleted from an Action List.

Entry Message Length

A number between 1 and 76 (minus the entry message position). Specifies (optionally) the length to which the list entry line is overlaid by the entry message, starting from the entry message position.

Present Empty List?

Determines whether the list is presented if it contains no items (YES or NO).

Auto Refresh Rate

This (optional) field determines whether a list is refreshed automatically by the system after a designated time period. Enter a time period in seconds (between 60 and 86400) representing the interval between a refresh of the list. If this field is left blank, then the list is not refreshed until the REFRESH key is pressed.

Note

For performance reasons, you should not use this setting unless there are special reasons for doing so; for example, a list showing the status of unresolved problems, that is used as a monitor, could be an effective use of this feature.

Heading Sub Char

This (optional) field determines whether the heading lines in the format contain variable data. Enter the character that precedes any variables defined in the heading lines.

Comments

A detailed description of the list's behavior. Although this is an optional field, it is recommended that comments be included to assist the administrator when maintaining list definitions.

After entering the list description details, press the FORWARD key to specify the list criteria on the CAS : List Criteria panel (as shown in Figure 8-3).

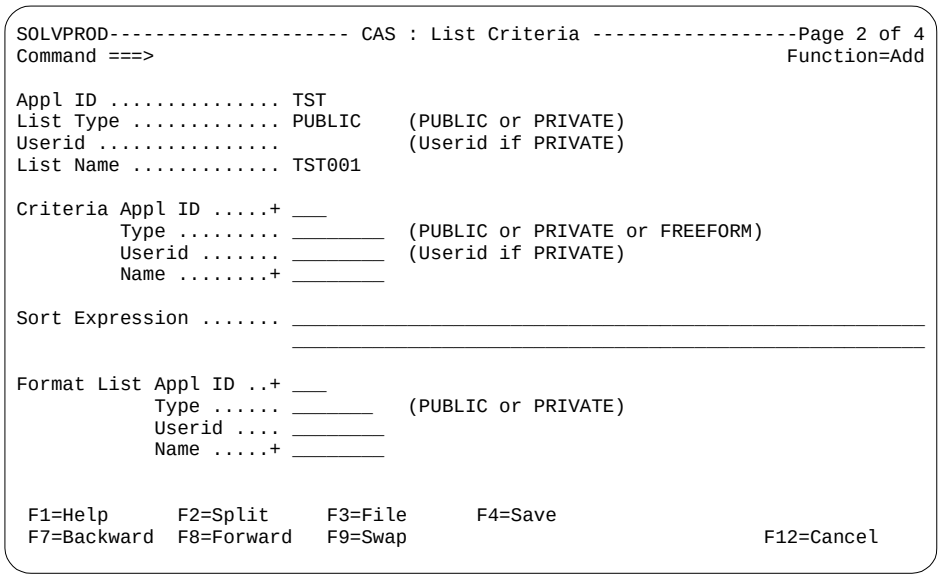


Figure 8-3. CAS : List Criteria Panel

The input fields displayed on the CAS : List Criteria panel are as follows:

Appl ID

The 3 character identifier of the application to which the list belongs.

List Type

A **List Type** is either:

- **PUBLIC** (available to all users)
- **PRIVATE** (available only to the owner of the list)

Userid

If a **List Type** is PRIVATE, the user ID of the owner must be entered (8 characters).

List Name

The name of a list is unique within an application and type. The name must be from 1 to 8 alphanumeric or national characters, or both.

Criteria Appl

The application ID of the criteria used to select items to go in the list.

Criteria Type

A **Criteria Type** can be either:

- **PUBLIC**, available to all users
- **PRIVATE**, available only to the owner of the criteria
- **FREEFORM,** allows the user to enter values for criteria on the criteria panel when the list is constructed

Criteria Userid

If a **Criteria Type** (see above) is **PRIVATE**, the user ID of the owner must be entered.

Criteria Name

The 1 to 8 character identifier of the criteria to be used when selecting items to be included in the list. If the **Criteria Type** is **PUBLIC** or **PRIVATE**, a **Criteria Name** must be entered.

Sort Expression

The **Sort Expression** (optionally) specifies a rule for sorting list entries by the service procedure before presentation. If this field is left blank, the entries are presented in the order in which they are returned by the service procedure.

The **Sort Expression** can contain constant or variable data, or both. Constant data is static, while variable data is set by the service procedure or list exit and substituted into the expression when the list is displayed.

The following fields allow you to specify a list definition from which the format, and entry line presentation attributes, for the current list are drawn when the list is displayed.

Format List Appl ID

Enter the application identifier of the list definition from which the format is drawn.

Type

Enter the type of the list; either PUBLIC or PRIVATE.

Userid

Enter the user ID of the owner of the list if the previous field contains PRIVATE.

Name

Enter the identifier of the list definition.

After entering the list criteria details, press the FORWARD key to display the CAS : Entry Line Presentation Attributes panel as shown in Figure 8-4.

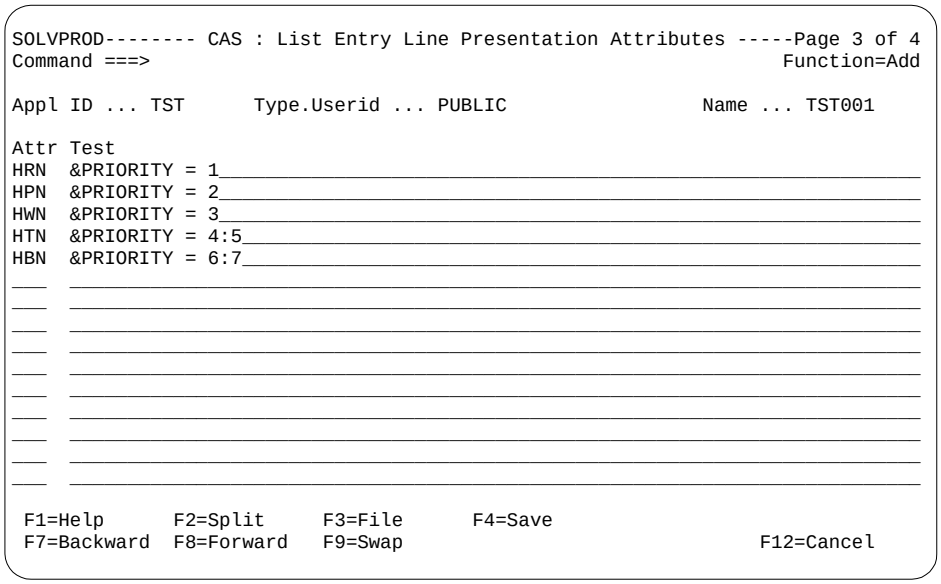


Figure 8-4. CAS : List Entry Line Presentation Attributes

You need to enter a code representing the presentation attributes and a matching boolean expression for the record that you want highlighted.

The fields displayed on the CAS : List Entry Presentation Attributes panel are as follows:

Attr

Enter a three character code representing the highlight attributes for the object. Prompting is available for valid highlighting attributes.

Test

Enter a boolean expression which is used to determine whether highlighting is applied to the list entry.

Enter up to 15 presentation attribute / boolean test pairs to highlight the specified entries within the list.

In the example shown in Figure 8-4, records with a priority of 1 are shown in high intensity red, with priority 2 are shown in high intensity pink, with priority 3 in high intensity white, with priority 4 or 5 in high intensity turquoise, and those with a priority of 6 or 7 in high intensity blue.

Note

If you specified that the format for the list be drawn from an existing list definition, you cannot define entry line presentation attributes.

After entering the Entry Line Presentation Attributes, press the FORWARD key to specify the list format on the CAS : List Format panel (as illustrated in Figure 8-5).

A text editor is provided; see Appendix B, *Text Editor Commands*, for the commands available.

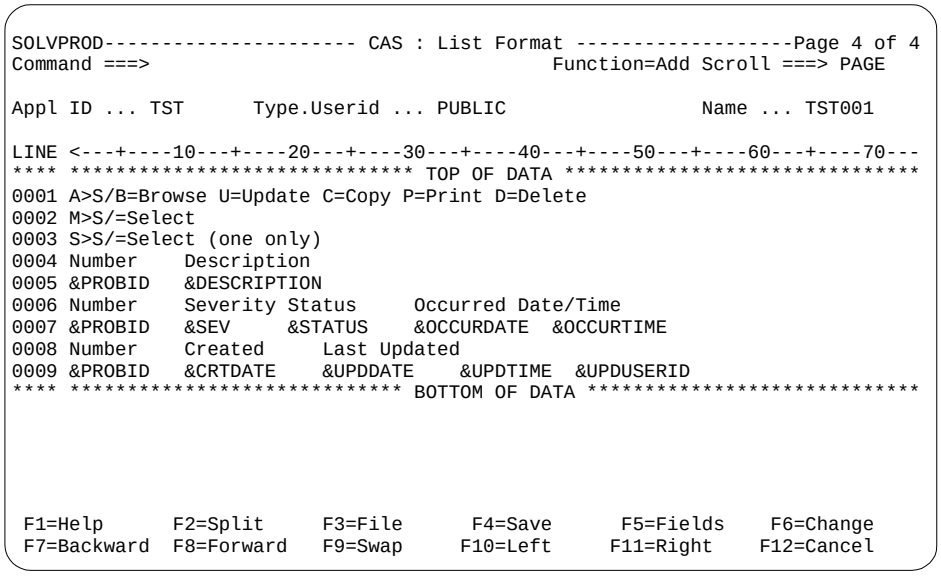


Figure 8-5. CAS : List Format Panel

At the top of the CAS : List Format panel text area, you can enter up to 4 comment lines, one for each type of list, to display specific instructions to the user. When the list is displayed, the appropriate comment line is shown on line 4 of the list panel. Comment lines are optional.

Begin Comment Line with...	**To define instructions for...**
A>	an Action List
M>	a Multiple Select List
S>	a Single Select List
N>	a Numbered List

Comment lines can contain NCL variables. For instance, you might want a comment line for an Action List to describe only those actions for which a user is authorized. You could achieve this by setting the entire comment line through a variable set by the list's service or exit procedure.

Any NCL variables defined in the comment line (except for system variables) must be set by the service procedure or list exit. The variable data in the comment line is then substituted by the system when a list is displayed using the list definition.

After entering the comment line(s), to display the actual list entries you must define one or more screen formats (up to a maximum of 10); for example, in Figure 8-5 three screen formats are defined. Each of these screen formats consists of a heading and an entry line.

Note

If you specified that the format for the list be drawn from an existing list definition, you cannot define a list format.

- A heading (up to 10 lines long) can contain constant or variable data, or both. Constant data is static, while variable data is set by the service procedure or list exit and substituted into the expression when the heading is displayed. For variables to be recognized, a **Heading Sub Char** must be specified; for example, if the NCL variable &USERID is used in the heading definition and the **Heading Sub Char** has been defined as &, the variable is replaced by the user's user ID when the heading is displayed.

  If you wanted the & to appear as a constant in the heading, you would have to define a different **Heading Sub Char**; therefore, if you specify the ~ character as the **Heading Sub Char**, for the previous example you would use ~USERID instead of &USERID.

- An entry line consists of entry line fields (NCL variables), each of which is replaced by an attribute of the list item when the list is displayed. An entry line can also contain constant data. Note, however, that the first non-blank character on the line must be an ampersand (&).

  Entry line fields can be:

  - Real variables
  - Aliases for real variables

  If you want to use aliases, press the FIELDS key to display the CAS : List Entry Line Fields panel (Figure 8-6). This panel displays all the entry line fields defined on the CAS : List Format panel.

```
SOLVPROD----------------- CAS : List Entry Line Fields -------------------------
 Command ===>                                      Function=Add Scroll ===> PAGE

 Appl ID ... TST      Type.Userid ... PUBLIC                Name ... 001

 Entry Line Field  Real Field
 A                 TSTID
 B                 TSTTYPE
 C                 TSTTIME
 D                 TSTDATE
 E                 TSTGROUP
 F                 NAME
 G                 PHONE
 CREDAT            CREDAT
 CRETIME           CRETIME
 CREUSER           CREUSER
 UPDTDATE          UPDTDATE
 UPDTTIME          UPDTTIME
 UPDTUSER          UPDTUSER
 **END**


  F1=Help      F2=Split     F3=File      F4=Save      F5=Format
  F7=Backward  F8=Forward   F9=Swap                              F12=Cancel
```

Figure 8-6. CAS : List Entry Line Fields Panel

The input fields displayed on the CAS : List Entry Line Fields panel are as follows:

Entry Line Field
The name of the field defined in an entry line.

Real Field
Enter the actual variable that corresponds to each entry line field. This variable is replaced by an attribute of the list entry when the list is displayed.

Make entries for each of the entry line field aliases that you have used.

After entering the list definitions, press the FILE key. To cancel the list specification, press the CANCEL key.

Maintaining List Definitions

Use the functions described in this section to maintain existing list definitions.

The functions described here can be selected from the CAS : List Definition Menu or can be selected as actions from a list of list definitions.

Listing List Definitions

Select option **L** from the CAS : List Definition Menu to produce a list of list definitions that are already defined.

If you enter values (or partial values) in the **Appl ID**, **List Type**, **Userid**, **List Name** and **Group** fields when you select this option, the list contains only those list definitions that match the values you have entered.

List definition in a list are displayed over five panels, as shown in Figure 8-7, Figure 8-8, Figure 8-9, Figure 8-10 and Figure 8-11. Use the RIGHT and LEFT keys to scroll between the panels.

```
SOLVPROD----------------- CAS : List Definition List --------------------------
Command ===>                                                 Scroll ===> PAGE

                                        S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Description                          File ID
   ZAM  PUB            ZAMAAALL  All Asset-Agreement Relaters         MODSUSR
   ZAM  PUB            ZAMAAFF   Freeform Asset-Agr Rel Search        MODSDIS
   ZAM  PUB            ZAMAAFMT  Asset-Agreement Relater Format       MODSDIS
   ZAM  PUB            ZAMAARLC  Related Asset-Agreement Relaters     MODSDIS
   ZAM  PUB            ZAMAARLP  Asset-Agreement Relater Prompt       MODSDIS
   ZAM  PUB            ZAMAASCH  Asset-Agreement Relater Search       MODSDIS
   ZAM  PUB            ZAMAA001  Asset-Agreement Relater Search       MODSDIS
   ZAM  PUB            ZAMAA002  Related Asset-Agreement Relaters     MODSDIS
   ZAM  PUB            ZAMAGALL  All Agreements                       MODSDIS
   ZAM  PUB            ZAMAGFF   Freeform Agreement Search            MODSDIS
   ZAM  PUB            ZAMAGFMT  Agreement Format                     MODSDIS
   ZAM  PUB            ZAMAGRLC  Related Agreements                   MODSDIS
   ZAM  PUB            ZAMAGRLP  Agreement Prompt                     MODSDIS
   ZAM  PUB            ZAMAGSCH  Agreement Search                     MODSDIS
   ZAM  PUB            ZAMAGSIM  Similar Agreements                  MODSDIS
   ZAM  PUB            ZAMAG001  Agreement by Asset Search           MODSDIS
   ZAM  PUB            ZAMAG002  Agreements                          MODSDIS
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 8-7. CAS : List Definition List (page 1)

```
SOLVPROD----------------- CAS : List Definition List --------------------------
Command ===>                                                 Scroll ===> PAGE

                                        S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Status    ------------Criteria-------------
   ZAM  PUB            ZAMAAALL  ACTIVE
   ZAM  PUB            ZAMAAFF   ACTIVE         FREEFORM
   ZAM  PUB            ZAMAAFMT  INACTIVE
   ZAM  PUB            ZAMAARLC  ACTIVE
   ZAM  PUB            ZAMAARLP  ACTIVE    ZAM  PUBLIC              ZAMAASCH
   ZAM  PUB            ZAMAASCH  ACTIVE    ZAM  PUBLIC              ZAMAASCH
   ZAM  PUB            ZAMAA001  ACTIVE    ZAM  PUBLIC              ZAMAASCH
   ZAM  PUB            ZAMAA002  ACTIVE
   ZAM  PUB            ZAMAGALL  ACTIVE
   ZAM  PUB            ZAMAGFF   ACTIVE         FREEFORM
   ZAM  PUB            ZAMAGFMT  INACTIVE
   ZAM  PUB            ZAMAGRLC  ACTIVE
   ZAM  PUB            ZAMAGRLP  ACTIVE    ZAM  PUBLIC              ZAMAGSCH
   ZAM  PUB            ZAMAGSCH  ACTIVE    ZAM  PUBLIC              ZAMAGSCH
   ZAM  PUB            ZAMAGSIM  ACTIVE    ZAM  PUBLIC              ZAMAGSIM
   ZAM  PUB            ZAMAG001  ACTIVE    ZAM  PUBLIC              ZAMAG001
   ZAM  PUB            ZAMAG002  ACTIVE    ZAM  PUBLIC              ZAMAG002
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 8-8. CAS : List Definition List (page 2)

```
SOLVPROD----------------- CAS : List Definition List --------------------------
Command ===>                                                 Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Group        ----------Format List-----------
   ZAM  PUB            ZAMAAALL  ZAMASSET     ZAM  PUBLIC             ZAMAAFMT
   ZAM  PUB            ZAMAAFF   ZAMASSET     ZAM  PUBLIC             ZAMAAFMT
   ZAM  PUB            ZAMAAFMT
   ZAM  PUB            ZAMAARLC                 ZAM  PUBLIC             ZAMAAFMT
   ZAM  PUB            ZAMAARLP                 ZAM  PUBLIC             ZAMAAFMT
   ZAM  PUB            ZAMAASCH  ZAMASSET     ZAM  PUBLIC             ZAMAAFMT
   ZAM  PUB            ZAMAA001  ZAMSERVICE   ZAM  PUBLIC             ZAMAAFMT
   ZAM  PUB            ZAMAA002                 ZAM  PUBLIC             ZAMAAFMT
   ZAM  PUB            ZAMAGALL  ZAMSERVICE   ZAM  PUBLIC             ZAMAGFMT
   ZAM  PUB            ZAMAGFF   ZAMSERVICE   ZAM  PUBLIC             ZAMAGFMT
   ZAM  PUB            ZAMAGFMT
   ZAM  PUB            ZAMAGRLC                 ZAM  PUBLIC             ZAMAGFMT
   ZAM  PUB            ZAMAGRLP                 ZAM  PUBLIC             ZAMAGFMT
   ZAM  PUB            ZAMAGSCH  ZAMSERVICE   ZAM  PUBLIC             ZAMAGFMT
   ZAM  PUB            ZAMAGSIM  ZAMAGR#REL   ZAM  PUBLIC             ZAMAGFMT
   ZAM  PUB            ZAMAG001  ZAMSERVICE   ZAM  PUBLIC             ZAMAGFMT
   ZAM  PUB            ZAMAG002  ZAMASSET#REL ZAM  PUBLIC             ZAMAGFMT
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 8-9. CAS : List Definition List (page 3)

```
SOLVPROD----------------- CAS : List Definition List --------------------------
Command ===>                                                 Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Service Proc  Exit Name  Data Source
   ZAM  PUB            ZAMAAALL  $OSLH84Z                 ZAMAAREL
   ZAM  PUB            ZAMAAFF   $OSLH84Z                 ZAMAAREL
   ZAM  PUB            ZAMAAFMT  $OSLH84Z
   ZAM  PUB            ZAMAARLC  $OSLH84Z                 ZAMAAREL
   ZAM  PUB            ZAMAARLP  $OSLH84Z                 ZAMAAREL
   ZAM  PUB            ZAMAASCH  $OSLH84Z                 ZAMAAREL
   ZAM  PUB            ZAMAA001  $OSLH84Z                 ZAMAAREL
   ZAM  PUB            ZAMAA002  $OSLH84Z                 ZAMAAREL
   ZAM  PUB            ZAMAGALL  $OSLH84Z                 ZAMAGR
   ZAM  PUB            ZAMAGFF   $OSLH84Z                 ZAMAGR
   ZAM  PUB            ZAMAGFMT  $OSLH84Z
   ZAM  PUB            ZAMAGRLC  $OSLH84Z                 ZAMAGR
   ZAM  PUB            ZAMAGRLP  $OSLH84Z                 ZAMAGR
   ZAM  PUB            ZAMAGSCH  $OSLH84Z                 ZAMAGR
   ZAM  PUB            ZAMAGSIM  $OSLH84Z                 ZAMAGR
   ZAM  PUB            ZAMAG001  $OSLH84Z                 ZAMAGR
   ZAM  PUB            ZAMAG002  $OSLH84Z                 ZAMAGR
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 8-10. CAS : List Definition List (page 4)

```
SOLVPROD----------------- CAS : List Definition List --------------------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Created     Last Updated
   ZAM  PUB            ZAMAAALL  20-FEB-1993 21-MAY-1993 11.40 USER01
   ZAM  PUB            ZAMAAFF   20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAAFMT  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAARLC  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAARLP  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAASCH  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAA001  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAA002  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGALL  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGFF   20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGFMT  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGRLC  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGRLP  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGSCH  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGSIM  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAG001  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAG002  20-FEB-1993 20-FEB-1993 00.00 INSTALL
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left
```

Figure 8-11. CAS : List Definition List (page 5)

The fields displayed on the CAS : List Definition List panels are as follows:

Appl
The application identifier of the list.

Typ
The list type; PUBLIC (PUB) or PRIVATE (PRI).

Userid
The user ID of the user to whom the list belongs if its type is PRIVATE.

Name
The identifier of the list

The above four fields are displayed on each screen of the List Definition List.

Description
A description of the list.

File ID
The identifier of the MODS control file where the list definition is held.

Status
Whether the list is ACTIVE or INACTIVE.

Criteria
If a criteria is defined for the list, this area displays (in order) the application identifier of the criteria, its type (public, or freeform), the user ID associated with the criteria if its type is private and the criteria identifier.

Group
The identifier of the group to which the list definition belongs (if defined).

Format List
If the list draws its format from another list definition, this area displays (in order) the application identifier of the list (from which the format is drawn), whether it is a public or private list, the user ID of the owner of the list (if it is private) and the identifier of the list definition.

Service Proc
The identifier of the service procedure for this list.

Exit Name
The identifier of the exit procedure for this list (if defined).

Data Source
The source of the data for this list.

Created
The date the list definition was created.

Last Updated
The date and time the list was last updated, and the user ID of the person who performed the update. If the list definition has not been modified since it was installed, INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B, S, or /	Browse the selected list definition
U	Update the selected list definition
D	Delete the selected list definition
C	Copy the selected list definition

These actions can also be executed from the CAS : List Definition Menu.

Browsing a List Definition

Option **B** displays the five panels of the list definition, as shown in Figure 8-2, Figure 8-3, Figure 8-4, Figure 8-5, and Figure 8-6 except that the function is Browse and the fields are displayed as output fields only.

Use the FORWARD and BACKWARD keys to move between the panels.

Updating a List Definition

Option **U** displays the five panels of the list definition, as shown in Figure 8-2, Figure 8-3, Figure 8-4, Figure 8-5, and Figure 8-6 except that the function is Update and the **Appl ID**, **List Type**, **Userid** and **List Name** fields are displayed as output fields.

Update the fields of the list definition as required; refer to the *Adding a List Definition* section for details.

After updating the list definition, press the FILE key. To cancel the update, press the CANCEL key.

Deleting a List Definition

Select option **D** to delete a list definition. This displays a message requiring you to confirm the deletion.

Press the ENTER key to delete the list definition, or press the CANCEL key to cancel the deletion.

Copying a List Definition

Select option **C** to copy an existing list definition to another (new) list definition.

The panels of the list definition are displayed with the fields set to the values defined in the selected list definition.

Modify these as required to differentiate this definition from the copied definition; refer to the section titled *Adding a List Definition* for field details.

After entering the new list definition, press the FILE key. To cancel the copy press the CANCEL key.

Resetting the List Cache

Select option **R** to reset the list cache.

This clears all list definitions from the cache. If the MODS file is shared the list cache needs to be reset on all systems except the system being used to maintain the list definitions. It also needs to be done if lists have been moved, copied, or deleted using the Definition Utility. Reset the list cache only *after* you have changed, moved, copied, or deleted list definitions on the maintenance system.

If you do not perform this action, the modified list definitions may not take effect on the other systems until the next time SOLVE management services is started up.

Note

You do not need to reset the list cache on the maintenance system after you delete or update a list definition. In either action, the list definition is deleted from the list cache on the maintenance system automatically (but *not* on other systems). The next time you access the list definition on the maintenance system, an updated copy is retrieved from the database.

9

Maintaining Panel Domains

A panel domain is a collection of panels which are grouped together for the purposes of data entry and other user interaction.

A panel domain specifies the panels that belong to it and the order in, and conditions under, which they are displayed.

This chapter describes how to maintain panel domain definitions.

About Panel Domains

A panel domain consists of a panel domain definition and a collection of element and path definitions that comprise all possible panels to which a transition can occur.

A panel domain definition contains identification information only. A domain definition must exist before you can define paths and elements within the domain.

The facilities for maintaining domain definitions are described in the section titled *Maintaining Panel Domain Definitions,* on page 9-5.

An element definition is a state defined within a panel domain. Each element definition contains the name of a panel or text attribute. The element name is used as the mnemonic to invoke the panel.

Panels corresponding to elements in a panel domain must already be defined (see Chapter 6, *Maintaining Panels,* for details of how to define panels).

When defining an element, you must specify the panel which corresponds to the element, and, optionally, a test which is used to determine the element's eligibility for display. You can also specify a criteria to be evaluated to determine an element's eligibility for display.

There can be up to 9999 elements in a domain. The facilities for maintaining element definitions are described in the section titled *Maintaining Element Definitions*, on page 9-9.

Text Panel Element

Within a panel domain an element can be defined as a panel on which freeform text is entered using a provided text editor.

These types of panels, called text panels, do not require a corresponding panel definition; they are created and maintained by the system. Define a text panel by specifying the panel as type TEXT and including the name of a text attribute (which is to contain the text entered by the user) as part of the element definition.

Panel Path Definitions

A panel path definition represents the logical connection between any two panels (elements) within a domain. Panel paths define the order in which panels are displayed when a user navigates a panel domain.

A panel path definition specifies the two elements connected by the path, and the weighting given to the path. There can be up to 9999 paths defined from/to any element.

The facilities for maintaining path definitions are described in the section titled *Maintaining Path Definitions*, on page 9-16.

Panel Domains

Panel domains are loaded at system initialization or whenever a panel domain is first referenced.

Whenever you make changes to a panel domain you must execute the Reload action so that the changes take effect. See *Reloading Panel Domains*, on page 9-9.

Defining a Panel Domain

You can define a panel domain by choosing the Add option from the CAS : Panel Domain Definition Menu (or by choosing the ADD action from a list of panel domain definitions) or by copying an existing panel domain definition. See *Copying a Panel Domain Definition*, on page 9-8.

Panel Domain Definition Menu

Panel domains are accessed through the CAS : Panel Domain Definition Menu (option **P** of the CAS : Maintenance Menu).

This panel allows you to add, browse, update, delete, copy or obtain a list of panel domain definitions.

The CAS : Panel Domain Definition Menu is illustrated in Figure 9-1.

Figure 9-1. CAS : Panel Domain Definition Menu

```
SOLVPROD------------ CAS : Panel Domain Definition Menu ----------------$PV010
Select Option ===>

   A   - Add Panel Domain
   B   - Browse Panel Domain
   U   - Update Panel Domain
   D   - Delete Panel Domain (and Elements/Paths)
   C   - Copy Panel Domain (and Elements/Paths)
   L   - List Panel Domains
   R   - Reload Panel Domain
   X   - Exit

Application ID ....+ ___      ( Required B U D C R  Optional A L )
Domain Type ........ _______  ( Required B U D C    Optional A L R )
Userid ............. ________ ( Optional all )
Domain Name ........ ________ ( Required B U D C    Optional A L R )




 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The input fields displayed on the CAS : Panel Domain Definition Menu are as follows:

Application ID

The 3 character identifier of the application to which the panel domain belongs.

Domain Type

The type is either:

- PUBLIC, available to all users
- PRIVATE, available only to the owner of the domain

Userid

If a domain type is PRIVATE, the user ID of the owner must be entered.

Domain Name

A (unique) 1 to 8 character identifier of the panel domain.

The following options are available on the CAS : Panel Domain Definition Menu:

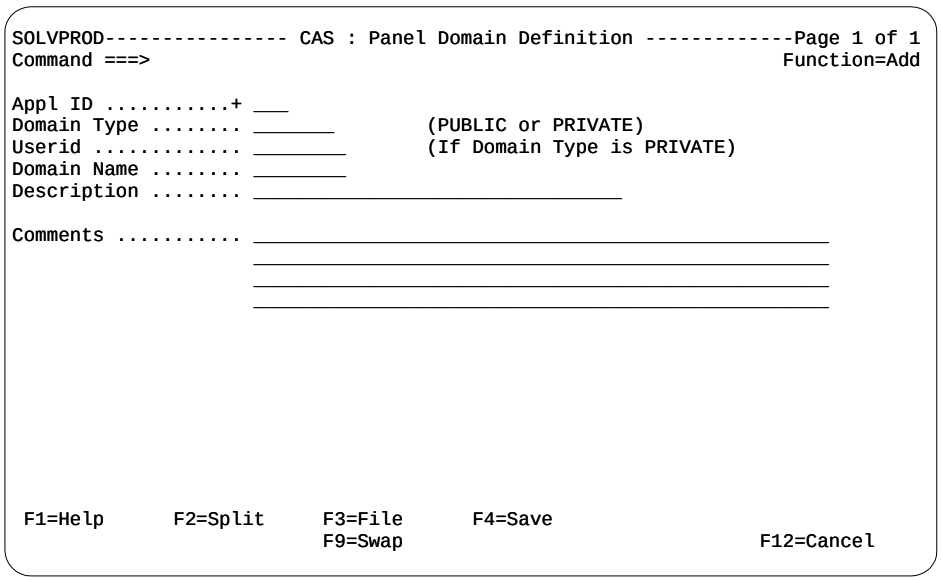
Option	Explanation
A	Add a new panel domain definition
B	Browse an existing panel domain definition
U	Update an existing panel domain definition
C	Copy an existing panel domain definition, including all elements and paths contained within it
D	Delete a panel domain definition, including all elements and paths contained within it
L	List panel domain definitions
R	Reload a panel domain

These options are discussed in the following sections.

Adding a Panel Domain Definition

To add a new panel domain definition select menu option **A.** The CAS : Panel Domain Definition panel is displayed, as shown in Figure 9-2.

Figure 9-2. CAS : Panel Domain Definition Panel



The fields displayed on the CAS : Panel Domain Definition panel are as follows:

Appl ID
The 3 character identifier of the application to which the panel domain belongs.

Domain Type
The type of domain. This is either:

- **PUBLIC** (available to all users)
- **PRIVATE** (available only to the owner of the domain)

Userid
If the domain type is **PRIVATE**, the user ID of the owner must be entered.

Domain Name
A (unique) 1 to 8 character identifier of the panel domain.

Description
This is a 1 to 32 character description of the panel domain. This is displayed when panel domains are listed for selection purposes, and should thus be as complete as possible.

Comments
A more detailed description of the panel domain. Although this is an optional field, it is recommended that comments be included to assist the administrator when maintaining panel domain definitions.

After specifying the panel domain, press the FILE key. To cancel the domain specification, press the CANCEL key.

Maintaining Panel Domain Definitions

This section describes maintenance options available for panel domain definitions. These options can be selected from the CAS : Panel Domain Definition Menu or applied as actions on a list of panel domain definitions.

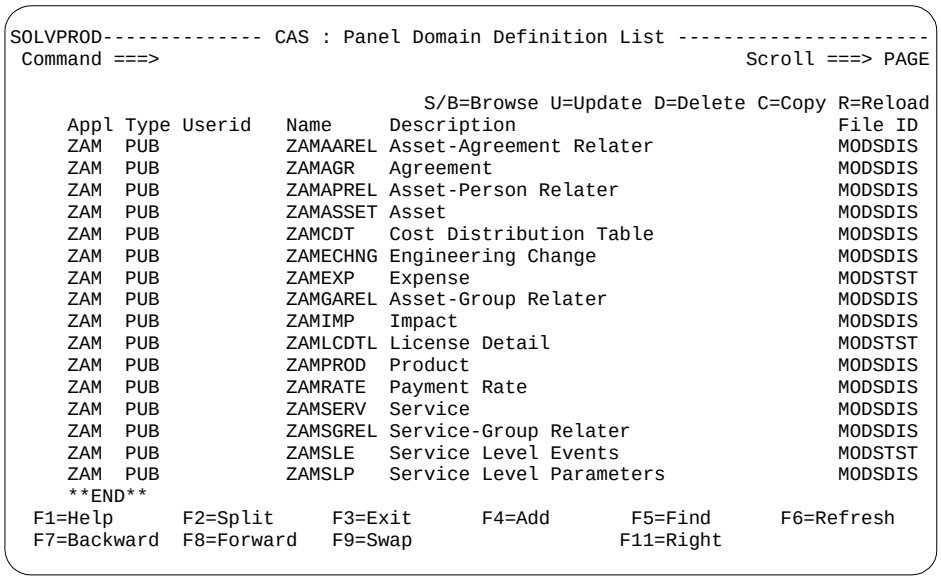
Listing Panel Domain Definitions

Select menu option **L** to display a list of panel domain definitions over two panels, as shown in Figure 9-3 and Figure 9-4.

If you make any entries in the fields displayed on the menu panel, definitions are displayed only if they match the values entered.

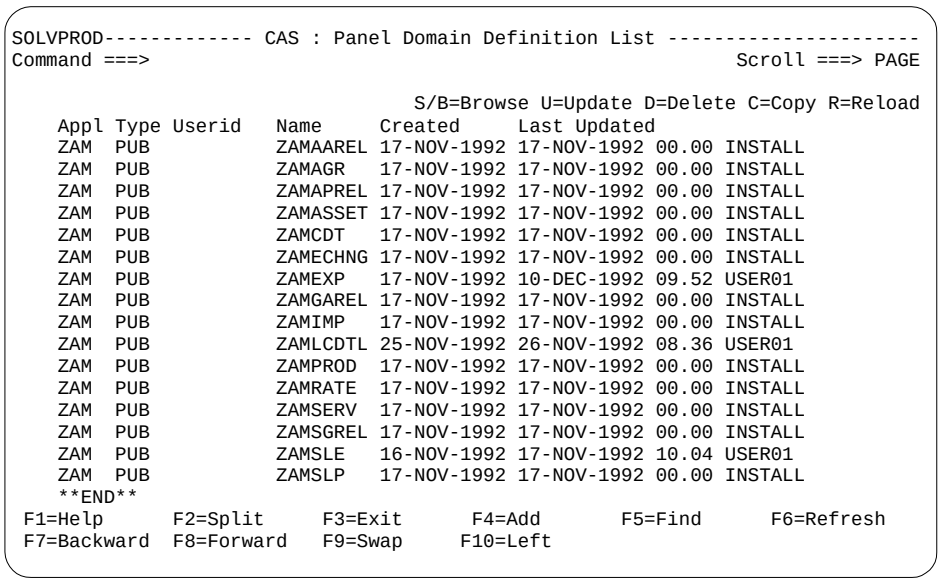
Use the RIGHT and LEFT keys to scroll between the panels.

Figure 9-3. CAS : Panel Domain Definition List (page 1)



```
SOLVPROD-------------- CAS : Panel Domain Definition List ----------------------
 Command ===>                                                  Scroll ===> PAGE

                                    S/B=Browse U=Update D=Delete C=Copy R=Reload
    Appl Type Userid   Name     Description                                File ID
    ZAM  PUB           ZAMAAREL Asset-Agreement Relater                    MODSDIS
    ZAM  PUB           ZAMAGR   Agreement                                  MODSDIS
    ZAM  PUB           ZAMAPREL Asset-Person Relater                       MODSDIS
    ZAM  PUB           ZAMASSET Asset                                      MODSDIS
    ZAM  PUB           ZAMCDT   Cost Distribution Table                    MODSDIS
    ZAM  PUB           ZAMECHNG Engineering Change                         MODSDIS
    ZAM  PUB           ZAMEXP   Expense                                    MODSTST
    ZAM  PUB           ZAMGAREL Asset-Group Relater                        MODSDIS
    ZAM  PUB           ZAMIMP   Impact                                     MODSDIS
    ZAM  PUB           ZAMLCDTL License Detail                             MODSTST
    ZAM  PUB           ZAMPROD  Product                                    MODSDIS
    ZAM  PUB           ZAMRATE  Payment Rate                               MODSDIS
    ZAM  PUB           ZAMSERV  Service                                    MODSDIS
    ZAM  PUB           ZAMSGREL Service-Group Relater                      MODSDIS
    ZAM  PUB           ZAMSLE   Service Level Events                       MODSTST
    ZAM  PUB           ZAMSLP   Service Level Parameters                   MODSDIS
    **END**
  F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
  F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 9-4. CAS : Panel Domain Definition List (page 2)



```
SOLVPROD------------- CAS : Panel Domain Definition List ----------------------
Command ===>                                                  Scroll ===> PAGE

                                  S/B=Browse U=Update D=Delete C=Copy R=Reload
   Appl Type Userid   Name     Created     Last Updated
   ZAM  PUB           ZAMAAREL 17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMAGR   17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMAPREL 17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMASSET 17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMCDT   17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMECHNG 17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMEXP   17-NOV-1992 10-DEC-1992 09.52 USER01
   ZAM  PUB           ZAMGAREL 17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMIMP   17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMLCDTL 25-NOV-1992 26-NOV-1992 08.36 USER01
   ZAM  PUB           ZAMPROD  17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMRATE  17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMSERV  17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMSGREL 17-NOV-1992 17-NOV-1992 00.00 INSTALL
   ZAM  PUB           ZAMSLE   16-NOV-1992 17-NOV-1992 10.04 USER01
   ZAM  PUB           ZAMSLP   17-NOV-1992 17-NOV-1992 00.00 INSTALL
   **END**
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left
```

The fields displayed on the CAS : Panel Domain Definition List are as follows:

Appl

The panel domain’s application identifier.

Type

The type of the definition; PRIVATE or PUBLIC.

UserID
The user ID of the owner of the definition if its type is PRIVATE.

Name
The identifier of the panel domain definition; the first three characters are the same as its application identifier.

Description
A brief description of the panel domain.

File ID
The identifier of the MODS control file where the panel domain definition is held.

Created
The date the panel domain definition was created.

Last Updated
Shows the date and time that the panel domain definition was last updated and the user ID of the person who performed the update. If the definition has not been updated since installation, INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item; to add a panel domain definition, press the ADD key.

Mnemonic	Action
B, S, or /	Browse the selected panel domain definition
U	Update the selected panel domain definition
D	Delete a panel domain definition (and all elements and paths contained within it)
C	Copy an existing panel domain definition, plus all elements and paths contained within it
R	Reload a panel domain

These actions perform the same function as the corresponding menu options.

Browsing a Panel Domain Definition

Select menu option **B** to display the CAS : Panel Domain Definition panel, as shown in Figure 9-2, except that in this case the function is Browse and the panel domain name cannot be modified.

You can list the domain's elements (see the CAS : Panel Domain Element Definition List, as shown in Figure 9-5 and Figure 9-7) and paths (as shown in the CAS : Panel Domain Path Definition list, Figure 9-9 and Figure 9-10) by pressing the PANEL and PATHS function keys respectively.

Updating a Panel Domain Definition

Select option **U** from the menu to display the CAS : Panel Domain Definition panel (as illustrated in Figure 9-2 on page 9-4). In this case the function is Update and the panel domain name cannot be modified.

Modify the domain details as required; refer to the section titled *Adding a Panel Domain Definition* for details.

You can list the domain's elements (see the CAS : Panel Domain Element Definition List, as shown in Figure 9-5 and Figure 9-7) and paths (as shown in the CAS : Panel Domain Path Definition List, Figure 9-9 and Figure 9-10) by using the PANEL and PATHS keys respectively.

Copying a Panel Domain Definition

Select menu option **C** to copy an existing panel domain definition, plus all its element definitions and path definitions, to another (new) panel domain definition.

When selecting option **C**, you must make entries in the following fields on the menu in order to specify the panel to copy:

- Application ID
- Domain Type
- Userid
- Domain Name

This option displays the CAS : Panel Domain Definition panel (as illustrated in Figure 9-2 on page 9-4) containing the new panel domain definition. The fields in the new panel domain definition contain the values of the copied panel domain definition.

Modify the domain details as required; refer to the section titled *Adding a Panel Domain Definition* for details.

After entering the new panel domain definition, press the FILE key. To cancel the copy, press the CANCEL key.

Deleting a Panel Domain Definition

Select menu option **D** to delete a panel domain definition, and all elements and paths contained within it.

When the delete confirmation message is displayed, press ENTER to delete the panel domain definition, element definitions and path definitions, or press the CANCEL key to cancel the deletion.

Reloading Panel Domains

Panel domains (and all elements and paths contained within them) are loaded into a VARTABLE cache for optimum performance when they are used.

Option **R** from the CAS : Panel Domain Definition Menu reloads a panel domain into the panel domain cache.

You must use this command whenever you make changes to a panel domain so that the modified panel domain is loaded and your changes take effect.

Maintaining Element Definitions

A panel domain has a series of elements associated with it that define the panels contained by it.

A domain's element definitions are accessed through the CAS : Panel Domain Element Definition List, as follows.

Listing Element Definitions

In order to list the element definitions for a panel domain you can do either of the following:

- Select either option **B** or option **U** from the CAS : Panel Domain Definition Menu and specify the identifier of the panel domain definition you want to add elements to. This displays the CAS : Panel Domain Definition panel (see Figure 9-2).
- Press the ELEMENTS function key. The CAS : Panel Domain Element Definition List is displayed as shown in Figure 9-5 to Figure 9-7. Use the RIGHT and LEFT keys to scroll between the panels.

Figure 9-5. CAS : Panel Domain Element Definition List (page 1)

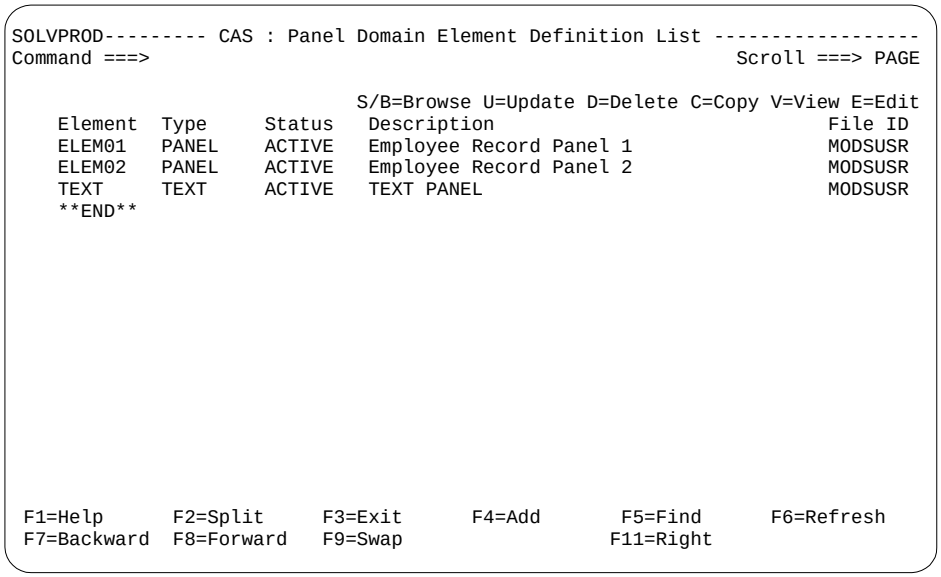


Figure 9-6. CAS : Panel Domain Element Definition List (page 2)

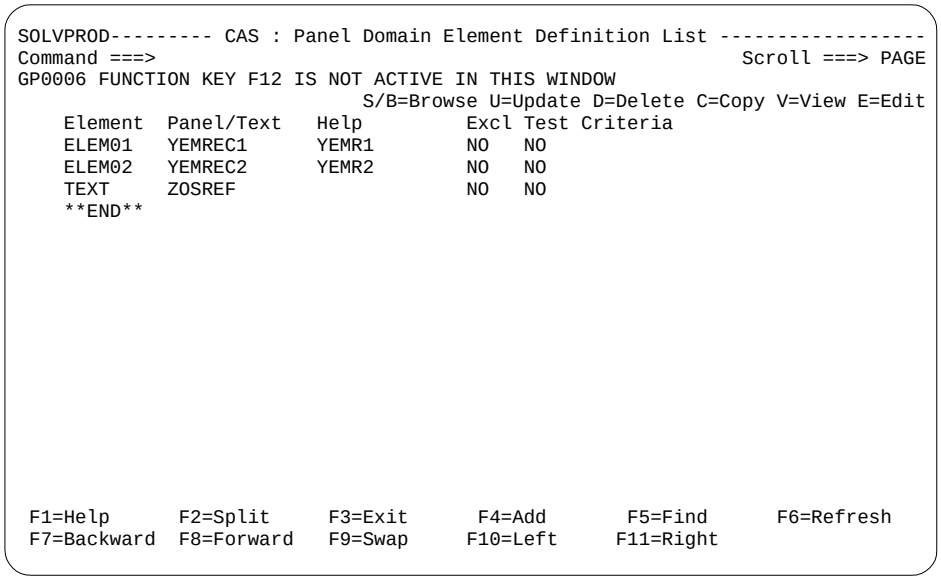
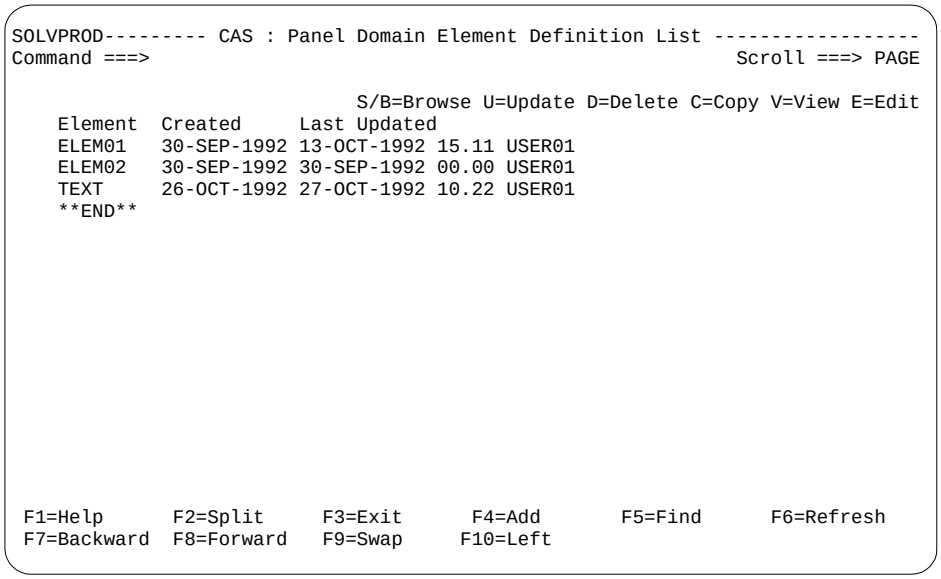


Figure 9-7. CAS : Panel Domain Element Definition List (page 3)



The fields displayed on the Panel Domain Element Definition List are as follows:

Element
The identifier of the element.

Type
The type of the element; PANEL or TEXT.

Status
The status of the panel; ACTIVE or INACTIVE.

Description
A brief description of the element.

File ID
The identifier of the MODS control file where the element is stored.

Panel/Text
The name of the panel or text attribute (for a text panel) represented by the element.

Help
The name of the help file associated with the element.

Excl
Indicates whether the element criteria is exclusive or not. If an element is exclusive, it overrides all other eligible elements of equal weight and is displayed in preference.

Test
YES if an eligibility test is specified.

Criteria
The name of the criteria if criteria is specified.

Created
The date the element definition was created.

Last Updated
The date and time the element was last updated and the user ID of the person who updated it. If the element has not been modified since installation, INSTALL is displayed.

You can perform the following actions on any element in the list, by placing the appropriate mnemonic next to it. To add a new element definition, press the ADD key.

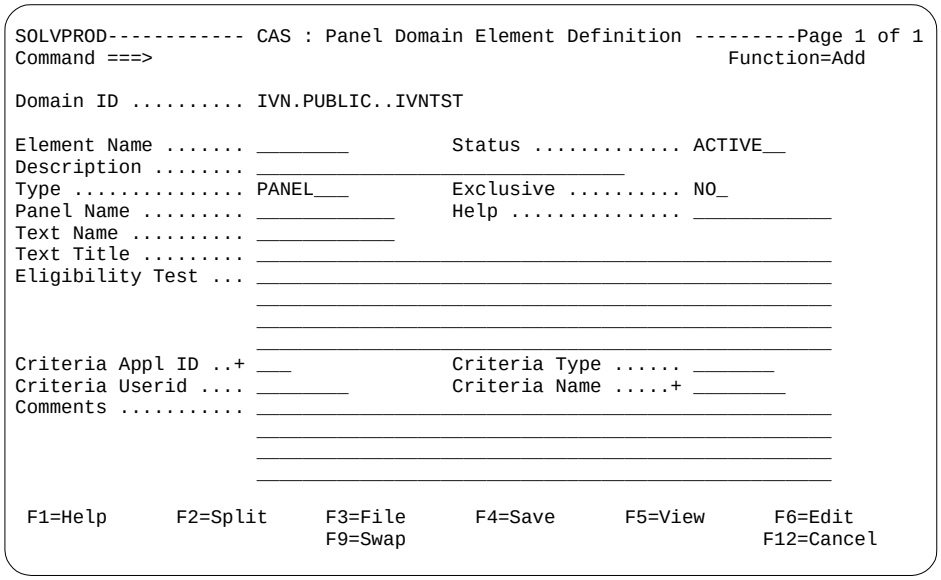
Mnemonic	Action
B, S, or /	Browse the selected element definition
U	Update the selected element definition
D	Delete the selected element definition
C	Copy the selected element definition
V	View the panel associated with the selected element (only for Type=PANEL)
E	Edit the panel associated with the selected element (only for Type=PANEL)

Detailed descriptions of these options follow.

Adding an Element Definition

To add a new element to the panel domain press the ADD key. This displays the CAS : Panel Domain Element Definition panel (as illustrated in Figure 9-8).

Figure 9-8. CAS : Panel Domain Element Definition Panel



The input fields displayed on the CAS : Panel Domain Element Definition Panel are as follows:

Element Name

The 8-character identifier of the element; must be unique within the domain.

Status

Flags the element as being either available for use (enter ACTIVE) or unavailable (enter INACTIVE).

Description

A 1- to 32-character description of the element. This field is displayed when elements are listed for selection purposes, and should thus be as full a description as possible.

Type

Indicates whether this element is associated with a data-entry panel defined using MODS : Panel Maintenance (enter PANEL), or with a text panel (enter TEXT).

- If you enter PANEL, you must specify a panel name in the **Panel Name** field.
- If you enter TEXT, you must specify an attribute in the **Text Name** field which holds the text, and you must specify the title of the panel in the **Text Title** field.

Exclusive

Specifies whether the element is exclusive or not. Enter **YES** or **NO**.

If an element is defined as being exclusive, it overrides all other eligible elements of equal weight and is displayed in preference.

Panel Name

The 12-character name of the panel associated with this element You must enter a panel name if the element type is PANEL.

Help

This field is optional for TYPE=TEXT but required for TYPE=PANEL. This field allows you to specify a function level help file to be displayed when the user requests help from within this element's panel.

If no help name is specified, the relevant application level help file is displayed.

Text Name

The 12 character name of the text attribute associated with this element. You must make an entry in this field if the element type is TEXT.

Text Title

The title line to be displayed at the top of the text panel. You must enter a text name if the element type is TEXT.

Eligibility Test

An optional boolean expression (up to 200 characters) used to test whether the element is eligible for display. The test can contain NCL variables to be substituted into the expression when required.

Criteria Appl ID

The application ID of the criteria used to determine whether the element is eligible for display.

Inclusion of a criteria in an element definition is optional. None or all of the **Criteria Appl ID**, **Criteria Type** and **Criteria Name** fields must be specified. For details of how to define a criteria, see Chapter 13, *Maintaining Criteria*.

Criteria Type

Criteria can be either PUBLIC or PRIVATE (inclusion of a criteria in an element definition is optional).

Criteria Userid

If a criteria type is PRIVATE, enter the user ID of the criteria's owner.

Criteria Name

The 8-character name of the criteria associated with this element.

Comments

A detailed description of the element. This field is optional.

After specifying the element, press the FILE key to add the new element definition. To cancel the element specification, press the CANCEL key.

Browsing an Element Definition

Select option **B** to display the CAS : Panel Domain Element Definition panel, as shown in Figure 9-8, in Browse mode.

Updating an Element Definition

Select option **U** to display the CAS : Panel Domain Element Definition panel (as illustrated in Figure 9-8).

Modify the element details as required; refer to the section titled *Adding an Element Definition*, on page 9-12 for details.

Deleting an Element Definition

Select option **D** to delete an element definition. A message is displayed requiring you to confirm the deletion. Press ENTER to delete the element, or press the CANCEL key to cancel the deletion.

Copying an Element Definition

Select option **C** to copy an existing element definition to another (new) element definition. This displays the CAS : Panel Domain Element Definition panel (as illustrated in Figure 9-8).

Update the new element details as required; refer to the section titled *Adding an Element Definition*, on page 9-12 for details.

After entering the new element definition, press the FILE key. To cancel the copy, press the CANCEL key.

Viewing an Element Definition

Select option **V** to display the panel associated with this element (that is, the panel specified in the element's **Panel Name** field), as it is displayed to the user.

This option is only valid for elements of type PANEL.

Editing an Element Definition

Enter **E** to display the panel associated with this element (that is, the panel specified in the element's **Panel Name** field), in Edit mode. You can edit the panel as described in Chapter 6, *Maintaining Panels*.

This option is only valid for elements of type PANEL.

Maintaining Path Definitions

A path definition specifies the order of display of a series of elements.

A domain's path definitions are accessed through the Path Definition List, as described below:

Listing Path Definitions

To access path definitions for a panel domain do the following:

1. Select either option **B** or option **U** from the CAS : Panel Domain Definition Menu (see Figure 9-1 on page 9-3). This displays the CAS : Panel Domain Definition panel (Figure 9-2).
2. Press the PATHS key. The Path Definition List is displayed as shown in Figure 9-9 and Figure 9-10. Use the RIGHT and LEFT commands (or the designated function keys) to scroll between the panels.

Figure 9-9. CAS : Panel Domain Path Definition List (page 1)

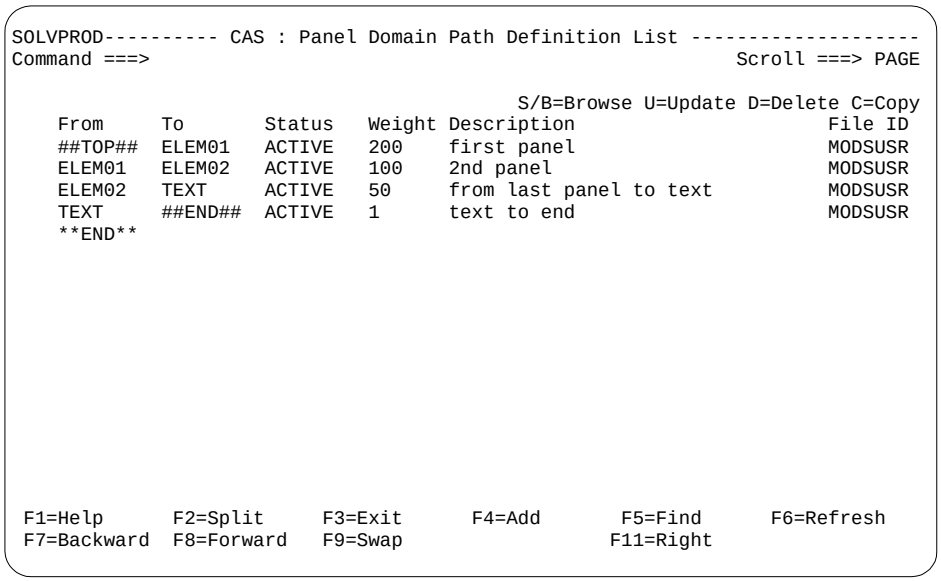
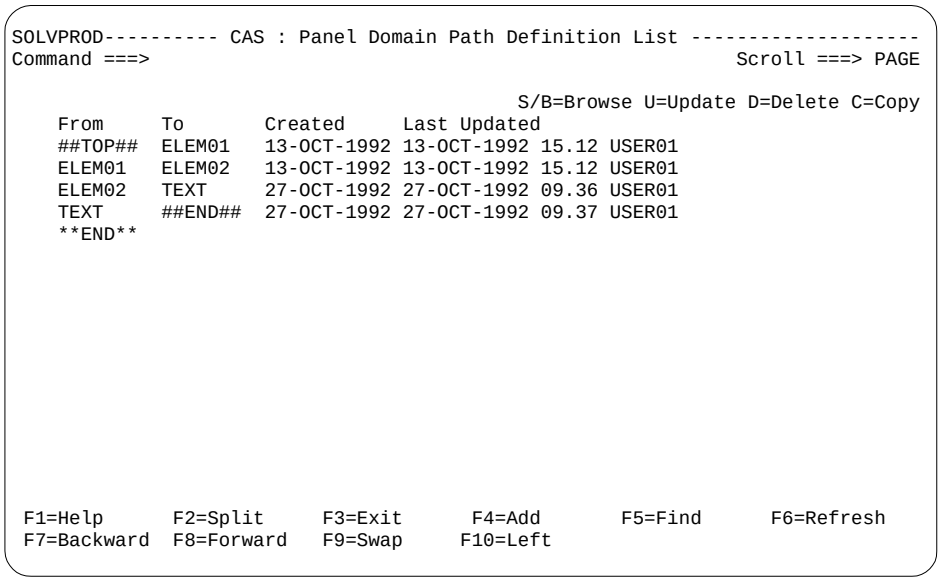


Figure 9-10. CAS : Panel Domain Path Definition List (page 2)



The fields displayed on the Panel Domain Path Definition List are as follows:

From

The identifier of the element the path goes from.

To

The identifier of the element the path goes to.

Status

Whether the element is ACTIVE or INACTIVE.

Description
A brief description of the path.

File ID
The identifier of the MODS control file where the path is held.

Created
The date the path was created.

Last Updated
The time and date the path was last updated and the user ID of the person who performed the update. If this path has not been updated since it was installed, INSTALL is displayed.

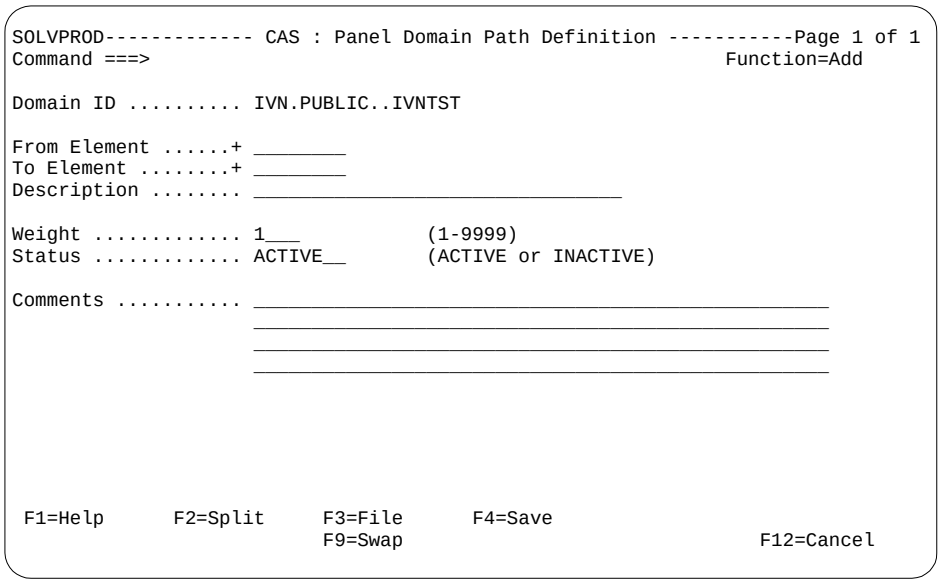
You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item. To add a new path definition, press F4.

Mnemonic	Action
B, S, or /	Browse the selected path definition
U	Update the selected path definition
D	Delete the selected path definition
C	Copy the selected path definition

Adding a Path Definition

To add a new path to the domain, access the CAS : Panel Domain Path Definition List (Figure 9-9 and Figure 9-10) and press the ADD function key. The CAS : Panel Domain Path Definition panel as illustrated in Figure 9-11 is displayed.

Figure 9-11. CAS : Panel Domain Path Definition Panel



The fields displayed on the CAS : Panel Domain Path Definition Panel are as follows:

From Element

The 8-character identifier of the element that the path goes from. The element must be defined in the domain.

If this is the entry path to the domain, the element must be named ##TOP##**.**

To Element

The 8-character identifier of the element that the path goes to. The element must be defined in the domain.

If this is the exit path from the domain, the element must be ##END##**.**

Description

A 1- to 32-character description of the panel path. This description is displayed when paths are listed for selection purposes, and should thus be as complete as possible.

Weight

A number (between 1 and 9999) used by CAS when determining the element to go to next. (Higher weighting takes precedence.)

Status

Flags the path as being either currently valid (ACTIVE) or not in use (INACTIVE).

Note

Any path which has either its ***To Element*** or its ***From Element*** defined as INACTIVE, is treated as INACTIVE.

Comments

A more detailed description of the path. This is an optional field.

After specifying the path, press the FILE key to add the new path definition. To cancel the path specification, press the CANCEL key.

Browsing a Path Definition

Enter **B** to display the CAS : Panel Domain Path Definition panel, as shown in Figure 9-11, in Browse mode.

Updating a Path Definition

Enter **U** to display the CAS : Panel Domain Path Definition panel (as illustrated in Figure 9-11). In this case the function is Update. Modify the path details as required; refer to the section titled *Adding a Path Definition*, on page 9-18 for details.

Deleting a Path Definition

Select option **D** to display a message requesting you to confirm the deletion. Press ENTER to delete the path, or press the CANCEL key to cancel the deletion.

Copying a Path Definition

Enter **C** to copy an existing path definition to another (new) path definition. This displays the CAS : Panel Domain Path Definition panel (as illustrated in Figure 9-11). Update the new path details as required; refer to the section titled *Adding a Path Definition*, on page 9-18 for details.

After entering the new path definition, press the FILE key. To cancel the copy, press the CANCEL key.

10

Maintaining Help

This chapter describes the Common Application Services (CAS) Help facility which is used to define and maintain online help for applications.

This facility provides a flexible means of defining context sensitive help at various application levels.

This chapter describes how to create and maintain online help files.

For a discussion of the structure of the Help Facility, including the Help Hierarchy, see the section titled *Help* in Chapter 2, *Concepts and Facilities*.

About Help

When a user presses the HELP key one or several panels are displayed which contain help information. The help that is displayed is dependent on where the user pressed the HELP key. For example, if the user requests help while the cursor is located in an input field then the help associated with that field is displayed.

Note

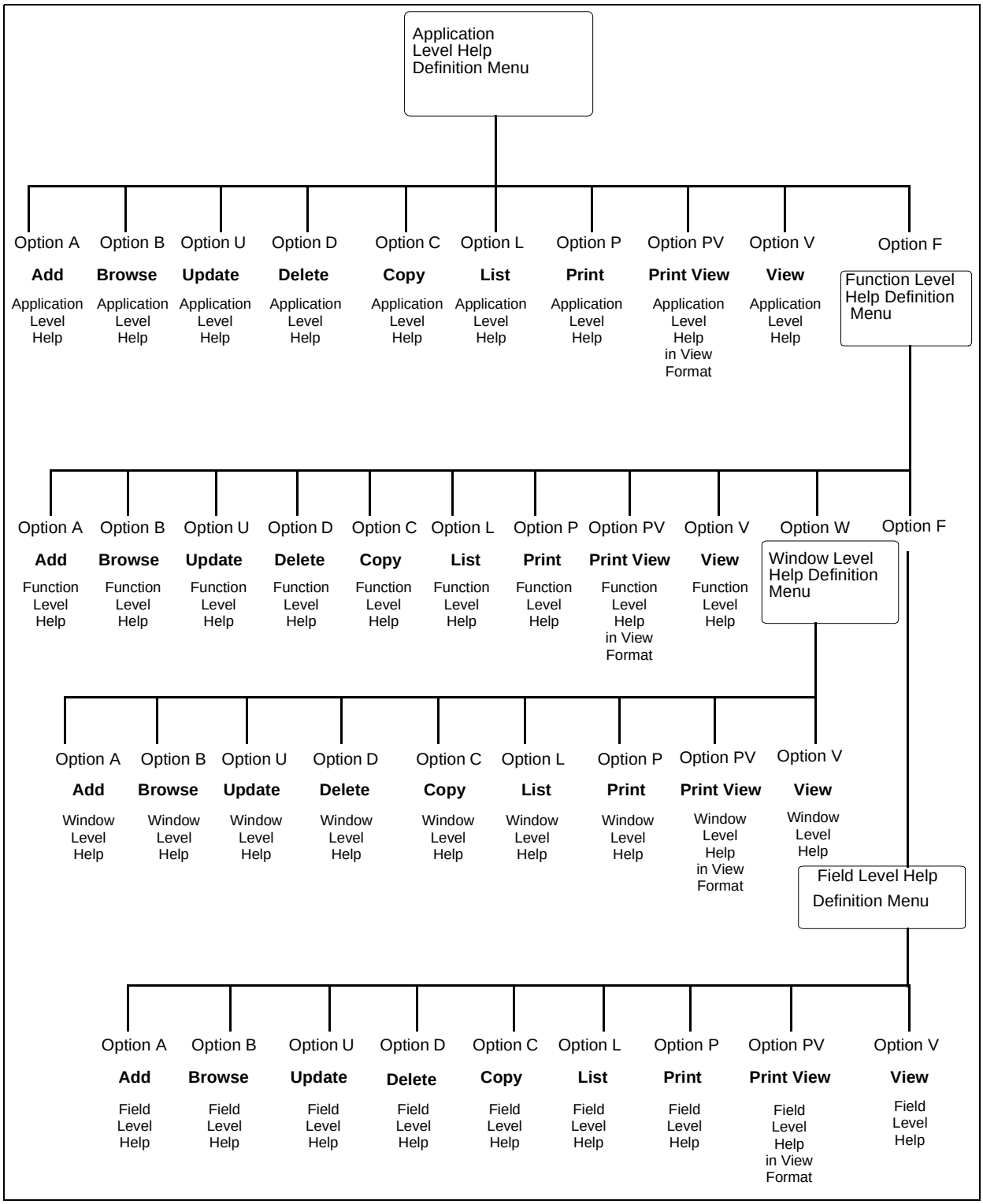
Not all fields are supported by the CAS Help facility.

Help text can be structured in panel format, in scrollable text format, or using a combination of both. Facilities for constructing menus, help indexes and tutorials within the help file are also available. Help files can be merged or copied, both during maintenance and while being displayed.

A text editor is available to define and maintain help text. Optional embedding of control codes is provided to make control of text color, highlight, intensity and formatting as easy as possible. Facilities for browsing (which displays the help file) and viewing (which displays the help as the user sees it) are available.

Help files can be added and maintained at either *application level*, *function level*, *window level* or *field level*, as illustrated in Figure 10-1.

Figure 10-1. Help Maintenance Panel Navigation



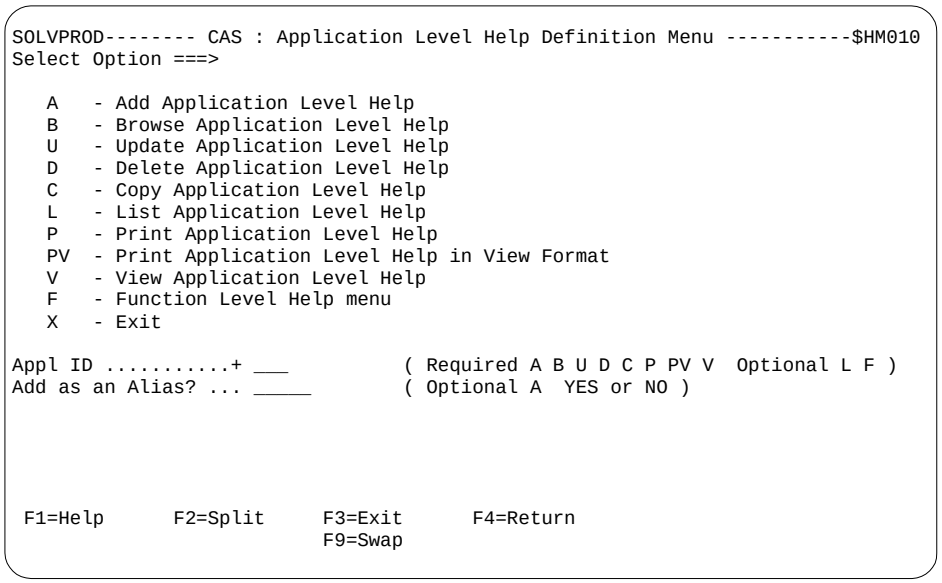
Maintaining Application Level Help

Application level help provides an overview of the application, its facilities and terminology. It can include a description of how to invoke a help tutorial or help index, if you include these in the help structure.

Application level help is accessed through the CAS : Application Level Help Definition Menu (option **H** of the CAS : Maintenance Menu).

This menu allows you to maintain application level help, or branch to the CAS : Function Level Help Definition Menu. The CAS : Application Level Help Definition Menu is illustrated in Figure 10-2.

Figure 10-2. CAS : Application Level Help Definition Menu



The input fields on the CAS : Application Level Help Definition Menu are as follows:

Appl ID

The 3 character identifier of the application to which the help file belongs.

Add as an Alias?

This field gives you the option of defining an additional name for an existing application help file. For details, see the section titled *Defining an Alias*, on page 10-25.

The following options are available on the CAS : Application Level Help Definition Menu:

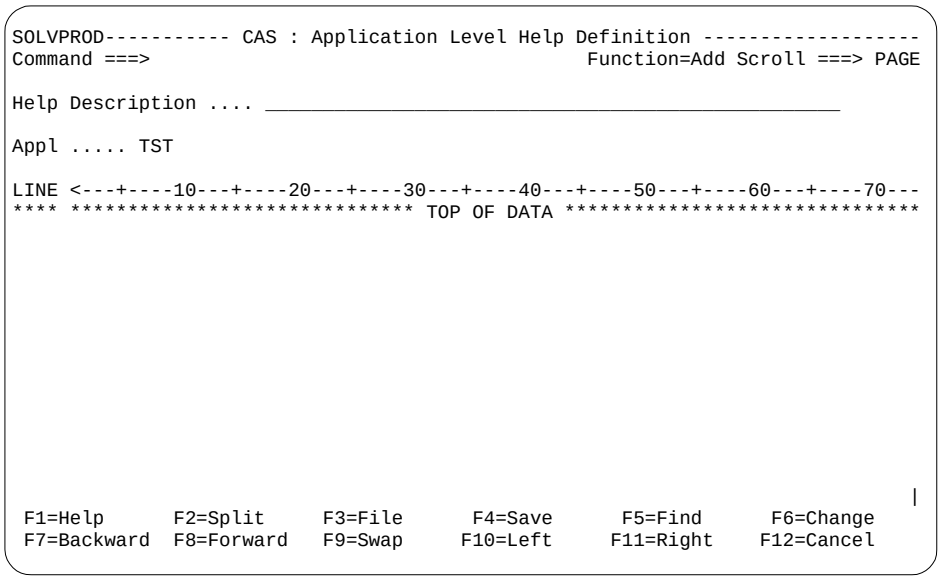
Option	Explanation
A	Add a new application-level help file
B	Browse an application-level help file
U	Update an application-level help file
D	Delete an application-level help file
C	Copy an application-level help file
L	List of application-level help files
P	Print application-level help files
PV	Print application-level help files in the format that a user sees when requesting help
V	View an application-level help file in the format that a user sees when requesting help
F	Go to the Function Level Help Definition Menu

A detailed description of these options follows.

Adding an Application Level Help File

Select option **A** to create a new help file, as shown in Figure 10-3.

Figure 10-3. CAS : Application Level Help Definition Panel



When you select this option you must specify the application identifier to which the help file belongs in the **Appl ID** field on the menu.

You can optionally specify the **Add as an Alias** field.

Before adding the help text, the following input field must be entered on the CAS : Application Level Help Definition Panel:

Help Description

An (up to) 50 character description of the help file. This description should state the name of the application to which this help file belongs, and give a brief summary of the file's contents.

The help text editor and formatting commands are available to you for adding text to a new help file; see *Facilities for Help Text Editing and Formatting*, on page 10-27 for details. Facilities for creating menus and selection lists are included.

After entering the help text, press the FILE key. To cancel the definition, press the CANCEL key.

Browsing an Application Level Help File

Option **B** displays the application-level help file that you selected (by making an entry in the **Appl ID** field on the menu). Use the FORWARD and BACKWARD keys to scroll the help file.

When specifying this option, the **Appl ID** field must contain the application identifier to which the help file belongs.

An example of a help file in Browse mode is shown in Figure 10-4.

Figure 10-4. CAS : Application Level Help File

```
SOLVPROD----------- CAS : Application Level Help Definition -------------------
Command ===>                                  Function=Browse Scroll ===> PAGE

Help Description .... CAS : Overview

Appl ..... $CA

LINE ----+----10---+----20---+----30---+----40---+----50---+----60---+----70---
**** ***************************** TOP OF DATA ******************************
0001 .TI CAS : Overview
0002 .PH C A S   O V E R V I E W
0003
0004 The Common Application Services (CAS) are a suite of application developme
0005 tools which are used to create and maintain application components such as
0006 panels, menus and lists.
0007
0008 CAS represents an easy-to-use, panel and menu driven means of customising
0009 applications. This approach to application maintenance means that you will
0010 rarely need to write NCL code to change an application unless you have
0011 specialized requirements outside the scope of CAS.
0012
0013 In general, when you want to make changes to the appearance and behavior
 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Updating an Application Level Help File

Option **U** displays the CAS : Application Level Help Definition panel, similar to the panel shown in Figure 10-4 except that the function is Update. Modify the help definition as required.

When specifying this option, the **Appl ID** field must contain the application identifier to which the help file belongs.

The help text editor and formatting commands are available to you when updating help files; see *Facilities for Help Text Editing and Formatting*, on page 10-27 for details. Facilities for creating menus are included.

After updating the help file, press the FILE key. To cancel the update, press the CANCEL key.

Deleting an Application Level Help File

Select option **D** to request the deletion of an application level help file. A delete confirmation message appears. Press ENTER to delete the application help file, or press the CANCEL key to cancel the deletion.

When specifying this option, the **Appl ID** field must contain the application identifier to which the help file belongs.

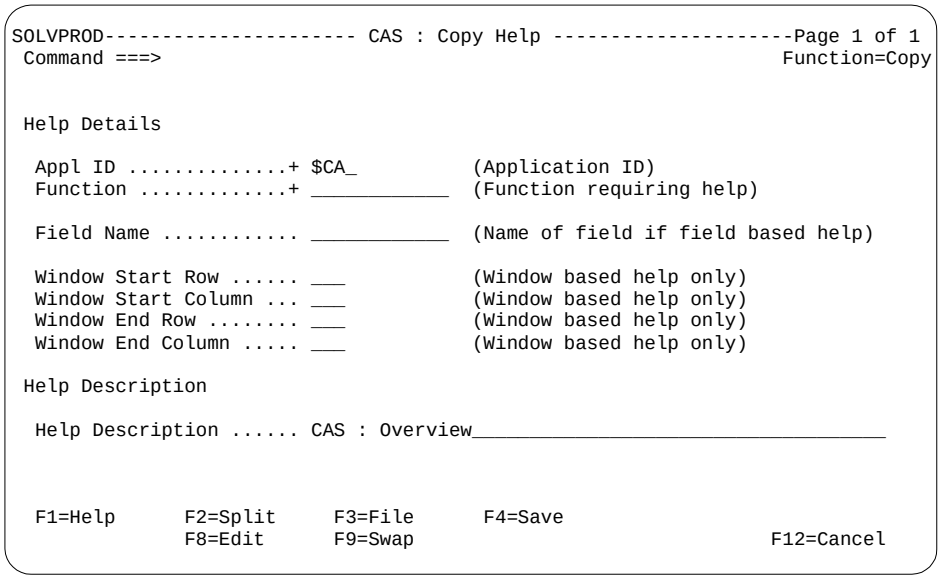
Press ENTER to delete the application-level help file, or press the CANCEL key to cancel the deletion.

Copying an Application Level Help File

To copy the contents of an existing application level help file to another (new) help file, select option **C** from the CAS : Application Level Help Definition Menu, and provide the application ID of the help file to copy from.

Option **C** displays the CAS : Copy Help panel, as shown in Figure 10-5. Enter the details of the new help file in this panel.

Figure 10-5. CAS : Copy Help Panel



The input fields displayed on the CAS : Copy Help Panel are as follows:

Appl ID

The 3-character identifier of the application to which the new help file belongs. This field is mandatory.

Function

The (up to) 12-character identifier of the function to which the new help file pertains. This field is required when the new file is function-level help, window-level help or field-level help.

Field Name

The (up to) 12-character name of the field which the new help file describes. This is only required when the new file is field-level help.

Window Start Row

The row of the screen on which the window starts; enter a number between 0 and 999. This field is only required when the new file is window-level help.

Window Start Column

The column of the screen on which the window starts; enter a number between 0 and 999. This field is only required when the new file is window-level help.

Window End Row

The row of the screen on which the window ends; enter a number between 0 and 999. This field is only required when the new file is window-level help.

Window End Column

The column of the screen on which the window ends; enter a number between 0 and 999. This field is only required when the new file is window-level help.

To edit the help text during the copy operation (before you file) enter the EDIT command or press the EDIT key.

To create the new help file, press the FILE key. To cancel the copy, press the CANCEL key.

Listing Application Level Help Files

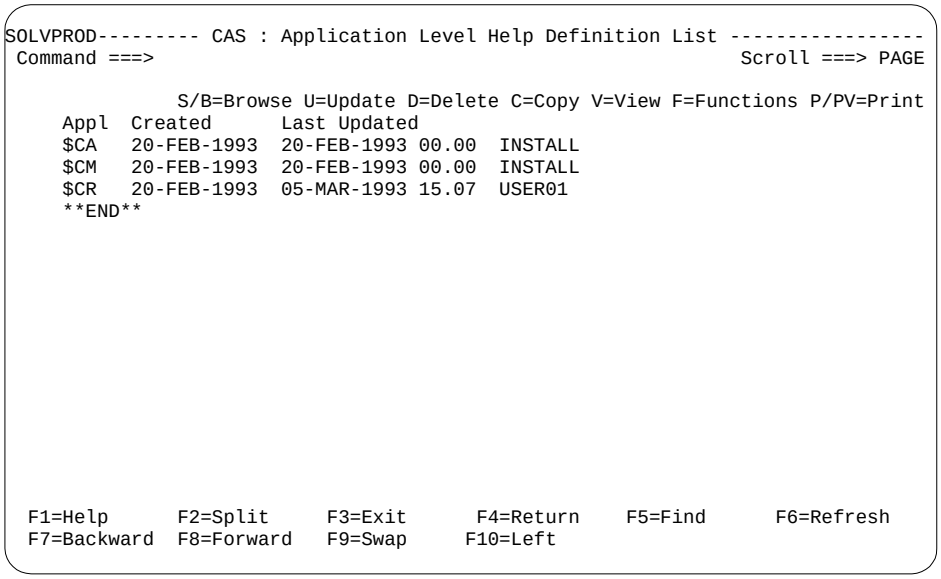
Option **L** displays a selection list of application-level help files, as shown in Figure 10-6 and Figure 10-7.

When specifying this option, you can optionally specify a prefix in the **Appl ID** field to limit the selection of records to be displayed on the list. For example, to list all application level help whose Appl ID starts with the $ character, specify $ in the **Appl ID** field.

Figure 10-6. CAS : Application Level Help Definition List (page 1)

```
SOLVPROD-------- CAS : Application Level Help Definition List -----------------
Command ===>                                                  Scroll ===> PAGE

             S/B=Browse U=Update D=Delete C=Copy V=View F=Functions P/PV=Print
    Appl  Description                                          Alias File ID
    $CA   CAS : Overview                                        No   MODSDIS
    $CM   CAS : Command Definition Overview                     No   MODSDIS
    $CR   CAS : Criteria Services Overview                      No   MODSUSR
    **END**

 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                 F11=Right
```

Figure 10-7. CAS : Application Level Help Definition List (page 2)



The fields displayed on the CAS : Application Level Help Definition List are as follows:

Appl
The application identifier of the help file.

Description
A description of the help file.

Alias
Whether an alias is defined for the help file (YES or NO).

File ID
The identifier of the MODS control file where the help file is maintained.

Created
The date the help file was created.

Last Updated
The time and date that the help file was last updated and the user ID of the person who performed the update. If the file has not been modified since installation, INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B, S, or /	Browse the selected application-level help file
U	Update the selected application-level help file
D	Delete the selected application-level help file
C	Copy the selected application-level help file
V	View the selected application-level help file
F	Display the function level help files associated with this application
P	Print the help file
PV	Print the help file in display format (as the user sees it)

Printing an Application Level Help File

To print the contents of an application level help file select option **P** from the CAS : Application Level Help Definition menu and enter the application ID in the **Appl ID** field on the menu.

Printing an Application Level Help File in View Format

Option **PV** is the same as the previous option except that the help file is printed in the format that a user sees when they request help.

Viewing an Application Level Help File

Option **V** displays the help file as the user sees it. Figure 10-8 shows the file shown in Figure 10-4, in display format.

Figure 10-8. Application Level Help File in Display Format

```
SOLVPROD---------------------- CAS : Overview ---------------------Page 1 of 8
Command ===>

                         +---------------------------+
                         |  C A S   O V E R V I E W  |
                         +---------------------------+

The Common Application Services (CAS) are a suite of application development
tools which are used to create and maintain application components such as
panels, menus and lists.

CAS represents an easy-to-use, panel and menu driven means of customising
applications. This approach to application maintenance means that you will
rarely need to write NCL code to change an application unless you have
specialized requirements outside the scope of CAS.

In general, when you want to make changes to the appearance and behavior of an
application you will most often use CAS to do so.

An overview of each of the components of CAS is presented below:


              F2=Split     F3=Exit      F4=Return                 F6=HelpHelp
              F8=Forward   F9=Swap                 F11=Index
```

Maintaining Function Level Help

Function level help describes a particular function within an application. Typically, each function-level help file is associated with one of the application's panels.

Function level help can also include:

- A *tutorial* for the application. A tutorial can consist of just a single help file, or it can refer to other help files through the .CP and .MU macros (see *Facilities for Help Text Editing and Formatting*, on page 10-27). If you define a function level help file called TUTORIAL, CAS automatically assigns a function key to access the tutorial in the Function Key Area on the screen, and display your tutorial file when the user presses that key.
- A *help index* — a menu of all help available for the application (use the .MU macro). If you define a function-level help file called INDEX, CAS automatically assigns a function key to access the index in the Function Key Area on the screen, and display your index file when the user presses that key.

  You have to write your own help index file. The help index file typically contains a number of .MU macros that display a menu of help topics which a user may select. For an example of a help index file, view the INDEX function level help for the $CA application ID.

Function level help is accessed through the CAS : Function Level Help Definition Menu (option **F** of the CAS : Application Level Help Definition Menu), as illustrated in Figure 10-9.

This panel allows you to maintain function-level help, or branch to the CAS : Window Level Help Definition Menu or the CAS : Field Level Help Definition Menu.

Figure 10-9. CAS : Function Level Help Definition Menu

```
SOLVPROD--------- CAS : Function Level Help Definition Menu ------------$HM020
Select Option ===>

   A   - Add Function Level Help
   B   - Browse Function Level Help
   U   - Update Function Level Help
   D   - Delete Function Level Help
   C   - Copy Function Level Help
   L   - List Function Level Help
   P   - Print Function Level Help
   PV  - Print Function Level Help in View Format
   V   - View Function Level Help
   W   - Window Level Help Definition Menu
   F   - Field Level Help Definition Menu
   X   - Exit

Appl ID ...........+              ( Required A B U D C L P PV V Optional F W )
Function Name .....+ _____________ ( Required A B U D C P PV V Optional L F W )
Add as an Alias? ... _____        ( Optional A  YES or NO )



 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The input fields displayed on the CAS : Function Level Help Definition Menu are as follows:

Appl ID

The 3 character identifier of the application to which the help file belongs.

Function Name

The (up to) 12 character name of the function which this help file describes.

Add as an Alias?

This field gives you the option of defining an additional name for an existing function-level help file. For details, see the section titled *Defining an Alias*, on page 10-25.

The following options are available on the Function Level Help menu:

Option	Explanation
A	Add a new function-level help file
B	Browse a function-level help file
U	Update a function-level help file
D	Delete a function-level help file
C	Copy a function-level help file
P	Print a function-level help file
PV	Print a function-level help file in the format that a user sees when requesting help
V	View a function-level help file in the format that a user sees when requesting help
L	List function-level help files
W	Go to the Window Level Help Definition Menu
F	Go to the Field Level Help Definition Menu

A detailed description of these options follows.

Adding a Function Level Help File

Option **A** opens the new function-level help file, similar to that shown in Figure 10-3. Enter a description of the help file in the **Help Description** field, then the text of the file (refer to the section *Adding an Application Level Help File*, on page 10-5 for details).

When specifying this option, the **Appl ID** and **Function Name** fields must also be specified.

You can also optionally specify the **Add as an Alias** field.

Browsing a Function Level Help File

Option **B** displays the function-level help file, similar to that shown in Figure 10-4. Use the FORWARD and BACKWARD keys to scroll the help file.

When specifying this option, the **Appl ID** and **Function Name** fields must also be specified.

Updating a Function Level Help File

Option **U** displays the CAS : Function Level Help Definition Panel, similar to that shown in Figure 10-4, except that the function is Update. Modify the file as required.

When specifying this option, the **Appl ID** and **Function Name** fields must also be specified.

Press the FILE key to save your changes. Press the CANCEL key to cancel changes that you have made.

Deleting a Function Level Help File

Select option **D** to delete a function level help file. A confirmation panel is displayed requiring you to confirm the deletion. Press ENTER to delete the help file, or press the CANCEL key to cancel the deletion.

When specifying this option, the **Appl ID** and **Function Name** fields must also be specified.

Copying a Function Level Help File

To copy the contents of an existing function-level help file to another (new) help file, select option **C** from the CAS : Function Level Help Definition Menu and provide the application ID and the function identifier of the help file to copy from.

When you do this the CAS : Copy Help panel is displayed, as shown in Figure 10-5. Enter the details of the new help file (as described in the section titled *Copying an Application Level Help File*, on page 10-7).

You can edit the help text in the copied file by pressing the EDIT key or entering EDIT in the command line.

To create the new help file, press the FILE key. To cancel the copy, press the CANCEL key.

Printing a Function Level Help File

To print the contents of a function-level help file, select option **P** from the CAS : Function Level Help Definition menu, and enter the application ID in the **Appl ID** field on the menu.

Printing a Function Level Help File in View Format

Option **PV** is the same as the previous option except that the help file is printed in the format that a user sees when they request help.

Viewing a Function Level Help File

When specifying this option, the **Appl ID** field must also be specified.

Option **V** displays a function-level help file as shown in Figure 10-8.

Listing Function Level Help Files

Option **L** displays a selection list of function-level help files, similar to that shown in Figure 10-6 and Figure 10-7.

When specifying this option, the **Appl ID** field must also be specified.

When specifying this option, you can optionally specify a prefix in the **Function Name** field to limit the selection of records to be displayed on the list. For example, to list all function-level help whose Function Name starts with the $ character, specify $ in the **Function Name** field.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B, S, or /	Browse the selected function-level help file
U	Update the selected function-level help file
D	Delete the selected function-level help file
C	Copy the selected function-level help file
V	View a function-level help file in the format that a user sees when requesting help
F	List the field level help files associated with this function level help
W	List the window level help files associated with this function level help
P	Print a function-level help file
PV	Print a function-level help file in the format that a user sees when requesting help

These functions can also be performed by choosing the corresponding option from the menu.

Maintaining Window Level Help

Window level help describes a particular window on a panel.

Window level help is accessed through the CAS : Window Level Help Definition Menu (option **W** of the CAS : Function Level Help Definition menu), as illustrated in Figure 10-10:

Figure 10-10. CAS : Window Level Help Definition Menu

```
SOLVPROD---------- CAS : Window Level Help Definition Menu -------------$HM030
Select Option ===>

   A   - Add Window Level Help
   B   - Browse Window Level Help
   U   - Update Window Level Help
   D   - Delete Window Level Help
   C   - Copy Window Level Help
   L   - List Window Level Help
   P   - Print Window Level Help
   PV  - Print Window Level Help in View Format
   V   - View Window Level Help
   X   - Exit

Appl ID ...........+ $CA          ( Required all )
Function Name .....+ _____________ ( Required all )
Start Row ......... ___           ( Required A B U D C P PV V  Optional L )
Start Column ....... ___          ( Required A B U D C P PV V  Optional L )
End Row ............ ___          ( Required A B U D C P PV V  Optional L )
End Column ......... ___          ( Required A B U D C P PV V  Optional L )
Add as an Alias? ... _____        ( Optional A  YES or NO )

 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The input fields displayed on the CAS : Window Level Help Definition Menu are as follows:

Appl ID

The 3-character identifier of the application. This is a mandatory field.

Function Name

The (up to) 12-character name of the function to which the window belongs. This is a mandatory field.

Start Row

The row of the screen on which the window starts (that is, the row that forms the top edge of the window). Enter a number between 0 and 999.

Start Column

The column of the screen on which the window starts (that is, the left side of the window). Enter a number between 0 and 999.

End Row

The row of the screen on which the window ends (that is, the row that forms the bottom edge of the window). Enter a number between 0 and 999.

End Column

The column of the screen on which the window ends (that is, the right side of the window). Enter a number between 0 and 999.

Add as an Alias?

This field gives you the option of defining an additional name for an existing window level help file. For details, see the section titled *Defining an Alias*, on page 10-25.

The following options are available on the Window Level Help menu:

Option	Explanation
A	Add a new window-level help file
B	Browse a window-level help file
U	Update a window-level help file
D	Delete a window-level help file
C	Copy a window-level help file
L	List window-level help files
P	Print a window-level help file
PV	Print a window-level help file in the format that a user sees when requesting help
V	View a window-level help file in the format that a user sees when requesting help

A detailed description of these options follows.

Adding a Window Level Help File

Option **A** opens the new window-level help file, similar to that shown in Figure 10-3 on page 10-5. Add a description of the help file, then the text of the file (refer to the section titled *Adding an Application Level Help File*, on page 10-5 for details).

When specifying this option, all fields on the menu except the **Add as an Alias** field, must be specified. The **Add as an Alias** field is optional.

Note

You cannot add window level help for a function until the function level help has been defined.

Browsing a Window Level Help File

Option **B** displays the window-level help file, similar to Figure 10-4. Use the BACKWARD and FORWARD keys to scroll the help file.

When specifying this option, all fields on the menu must be specified except the **Add as an Alias** field, which should not be specified.

Updating a Window Level Help File

Option **U** displays to the CAS : Window Level Help Definition Panel, similar to that shown in Figure 10-4, except that the function is Update. Modify the file as required.

When specifying this option, all fields on the menu must be specified except the **Add as an Alias** field, which should not be specified.

Deleting a Window Level Help File

Select option **D** to delete a window level help file. This displays a confirmation panel requiring you to confirm the deletion.

When specifying this option, all fields on the menu must be specified except the **Add as an Alias** field, which should not be specified.

Press ENTER to delete the help file, or press the CANCEL key to cancel the deletion.

Copying a Window Level Help File

To copy the contents of an existing window-level help file to another (new) help file, select option **C** from the CAS : Window Level Help Definition Menu, and provide the following details of the help file to copy from:

- Appl ID
- Function Name
- Start Row
- Start Column
- End Row
- End Column

This option displays the CAS : Copy Help Panel, as shown in Figure 10-5. This panel requires details of the new help file (as described in the section titled *Copying an Application Level Help File*, on page 10-7).

You can edit the help text in the copy before it is filed by pressing the EDIT key.

When you have finished, press the FILE key to create the new help file. To cancel the copy, press the CANCEL key.

Listing Window Level Help Files

Option **L** displays a selection list of window-level help files, similar to that shown in Figure 10-6 and Figure 10-7.

When specifying this option, the **Appl ID** and **Function Name** fields must be specified.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B	Browse the selected window-level help file
U	Update the selected window-level help file
D	Delete the selected window-level help file
C	Copy the selected window-level help file
P	Print the selected window level help file
PV	Print the selected window-level help file in the format that a user sees when requesting help
V, S, or /	View the selected window-level help file in the format that a user sees when requesting help

These actions operate in the same way as the corresponding menu options.

Printing a Window Level Help File

To print the contents of a window level help file, select option **P** from the CAS : Window Level Help Definition menu, and enter the application ID in the **Appl ID** field, and the function name in the **Function Name** field on the menu. Also enter the **Start Row**, **Start Column**, **End Row**, and **End Column** fields.

Printing a Window Level Help File in View Format

Option **PV** is the same as the previous option except that the help file is printed in the format that a user sees when they request help.

Viewing a Window Level Help File

Option **V** displays a window-level help file as the user sees it. A sample help file is shown in Figure 10-8 on page 10-12.

Maintaining Field Level Help

Field level help defines help for a specified field on a panel.

Field level help is accessed through the CAS : Field Level Help Definition Menu (option **F** from the Function Level Help Menu), as illustrated in Figure 10-11:

Figure 10-11. CAS : Field Level Help Definition Menu

```
SOLVPROD---------- CAS : Field Level Help Definition Menu -------------$HM040
 Select Option ===>

    A   - Add Field Level Help
    B   - Browse Field Level Help
    U   - Update Field Level Help
    D   - Delete Field Level Help
    C   - Copy Field Level Help
    L   - List Field Level Help
    P   - Print Field Level Help
    PV  - Print Field Level Help in View Format
    V   - View Field Level Help
    X   - Exit

 Appl ID ...........+ ___           ( Required all )
 Function Name .....+ _____________ ( Required all )
 Field Name ......... _____________ ( Required A B U D C P PV V  Optional L )
 Add as an Alias? ... _____        ( Optional A  YES or NO )




  F1=Help      F2=Split     F3=Exit      F4=Return
                            F9=Swap
```

The input fields displayed on the CAS : Field Level Help Definition Menu are as follows:

Appl ID

The 3-character identifier of the application. This field is mandatory.

Function Name

The (up to) 12-character name of the function to which the field belongs. This field is mandatory.

Field Name

The (up to) 12-character name of the field which this help file describes.

Add as an Alias?

This field gives you the option of defining an additional name for an existing field-level help file. For details, see the section titled *Defining an Alias*, on page 10-25.

The following options are available on the Field Level Help Menu:

Option	Explanation
A	Add a new field-level help file
B	Browse a field-level help file
U	Update a field-level help file
D	Delete a field-level help file
C	Copy a field-level help file
L	List field-level help files
P	Print the selected window level help file
PV	Print the selected field-level help file in the format that a user sees when requesting help
V	View a field-level help file in the format that a user sees when requesting help

A detailed description of these options follows.

Adding a Field Level Help File

Option **A** opens the new field-level help file, similar to that shown in Figure 10-3. Add a description of the help file and then the text of the file (refer to the section titled *Adding an Application Level Help File*, on page 10-5 for details).

When specifying this option, all fields on the menu except the **Add as an Alias** field, must be specified. The **Add as an Alias** field is optional.

You cannot add field level help for a function until function level help has been defined.

Browsing a Field Level Help File

Option **B** displays the field-level help file, similar to Figure 10-4, except that the function is Browse. Use the FORWARD and BACKWARD keys to scroll the help file.

When specifying this option, all fields on the menu except the **Add as an Alias** field which should not be specified.

Updating a Field Level Help File

Option **U** displays the CAS : Field Level Help Definition Panel, similar to that shown in Figure 10-4, except that the function is Update. Modify the file as required (refer to the section titled *Adding an Application Level Help File*, on page 10-5 for details).

When specifying this option, all fields on the menu except the **Add as an Alias** field which should not be specified.

Deleting a Field Level Help File

Select option **D** to delete a field level help file. When you do this a message is displayed so that you can confirm the deletion. Press ENTER to confirm the deletion or the CANCEL key to cancel the operation.

When specifying this option, all fields on the menu except the **Add as an Alias** field which should not be specified.

Copying a Field Level Help File

To copy the contents of an existing field-level help file to another (new) help file, select option **C** from the Field Level Help menu, providing the following details of the help file to copy from:

- Appl ID
- Function Name
- Field Name

This displays the Copy Help Panel, as shown in Figure 10-5. This panel requires details of the new help file (as described in the section titled *Copying an Application Level Help File*, on page 10-7).

You can edit the help text in the copy before it is filed by pressing the EDIT key.

To create the new help file, press the FILE key. To cancel the copy, press the CANCEL key.

Listing Field Level Help Files

Option **L** displays a selection list of field-level help files, similar to that shown in Figure 10-6 and Figure 10-7.

When specifying this option, the **Appl ID** and **Function Name** fields must also be specified. You can optionally specify a prefix in the **Field Name** field to limit the selection of records to be displayed on the list. For example, to list all field-level help whose Field Name starts with the $ character, specify $ in the **Field Name** field.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B, S, or /	Browse the selected field-level help file
U	Update the selected field-level help file
D	Delete the selected field-level help file
C	Copy the selected field-level help file
P	Print the selected field-level help file
PV	Print the selected field-level help file in the format that a user sees when requesting help
V	View the selected field-level help file in the format that a user sees when requesting help

These actions can also be selected by choosing the corresponding menu options.

Printing a Field Level Help File

To print the contents of a field level help file, select option **P** from the CAS : Window Level Help Definition menu. Enter the application ID in the **Appl ID** field, enter the function name in the **Function Name** field, and enter the identifier of the field in the **Field Name** field.

Printing a Field Level Help File in View Format

Option **PV** is the same as the previous option except that the help file is printed in the format that a user sees when they request help.

Viewing a Field Level Help File

Option V displays a field-level help file as the user sees it. A sample help file in display format is shown in Figure 10-8 on page 10-12.

Defining an Alias

An alias is a pointer to an existing help file. It specifies a different name for the file so that it can be re-used. Creating an alias, however, does not create a second file. Aliases represent an efficient means of using help files more than once.

When you define an alias you must specify the new help file and then specify the existing help file to be aliased.

To define an alias, select option **A** from any of the Help Definition Menus (shown in Figure 10-2, Figure 10-9, Figure 10-10, and Figure 10-11).

- Enter the **Appl ID** of the alias help, (not the existing help.)
- Depending on the type of alias you are defining, fill in any other fields required to perform an Add. (These values are for the alias, not the existing help.)
- Enter **YES** in the field ***Add as an Alias?***

Press ENTER to display the CAS : Select Alias Help Panel, as shown in Figure 10-12.

Figure 10-12. CAS : Select Alias Help Panel

```
SOLVPROD------------------ CAS : Select Alias Help -----------------Page 1 of 1
 Command ===>                                                  Function=ADD


 Alias Help Details

  Appl       Func              Field              Window

  Alias Description ...... ____________________________________________________

 Actual Help Details

  Appl ID ..............+ ____          (Application ID)
  Function .............+ ____________  (Function requiring help)

  Field name ............ ____________  (Name of field if field based help)

  Window Start Row ...... ___           (Window based help only)
  Window Start Column ... ___           (Window based help only)
  Window End Row ........ ___           (Window based help only)
  Window End Column ..... ___           (Window based help only)

  F1=Help      F2=Split     F3=File
                            F9=Swap                 F11=List     F12=Cancel
```

The details of the alias, which you entered on the previous menu, are shown in the **Alias Help Details** fields on the CAS : Select Alias Help Panel.

The fields displayed on the CAS : Select Alias Help Panel are as follows:

Alias Description

An (up to) 50-character description of the alias help file.

You now need to provide details of the existing help file for which you want to define an alias. You can do this by either:

- Entering the help file's details directly into the the **Actual Help Details** fields (**Appl ID**, **Function**, **Field name** and the **Window** fields as required).
- Pressing the LIST key to display a list of all existing help files, from which you can select the file you require. The file's details are copied into the **Actual Help Details** fields.

  The CAS : Select Alias Help List is shown in Figure 10-13.

Figure 10-13. CAS : Select Alias Help List

```
SOLVPROD---------------- CAS : Select Alias Help List -------------------------
Command ===>                                                 Scroll ===> PAGE

                                                                      S=Select
                      Help
   Appl  Function     Type    Description                              File ID
   $BS   INDEX        Func    Help Index                               MODSDIS
   $CA                Appl    CAS : Overview                           MODSDIS
   $CA   INDEX        Func    CAS : Help Index                         MODSDIS
   $CA   MENU010      Func    CAS : Maintenance Menu                   MODSDIS
   $CM                Appl    CAS : Command Definition Overview        MODSUSR
   $CM   CMDENTRY     Func    CAS : Command Definition Entry           MODSUSR
   $CM   CMDLIST      Func    CAS : Command Definition List            MODSUSR
   $CM   INDEX        Func    CAS : Command Definition Maintenance I   MODSUSR
   $CM   MENU010      Func    CAS : Command Definition Menu            MODSUSR
   $CR                Appl    CAS : Criteria Definition Overview       MODSUSR
   $CR   CRITERIA     Func    Criteria                                 MODSUSR
   $CR   DATACRITCRIT Func    Criteria Text                            MODSUSR
   $CR   DATACRITDESC Func    Criteria Description                     MODSUSR
   $CR   DATACRITPARM Func    Criteria Exit Parameters                 MODSUSR
   $CR   INDEX        Func    Criteria Definition Help Index           MODSUSR
   $CR   LISTCRITDEF  Func    Criteria Definition List                 MODSUSR
F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Select the required help file by placing an **S** next to it and pressing ENTER. This transfers you back to the CAS : Select Alias Help Panel, with the appropriate fields completed with details of the help file you have selected.

From the CAS : Select Alias Help Panel, press FILE to create the alias help file, or the CANCEL key to cancel the alias.

Facilities for Help Text Editing and Formatting

The text contained within help files can be modified using the help text editor. A number of text formatting macros are available; by embedding these in the help text you can control the appearance of the text when displayed to the user.

Help Text Editor

A text editor is available to edit the help files. See Appendix B, *Text Editor Commands*, for a full description of this facility.

Help Macros

Help macros can be embedded in help files to control the appearance of text when displayed on a panel. They must start in column 1, and there can be only one macro per line. The macros are summarized below; full descriptions follow.

Table 10-1. Help Macros

Command	Action
.AT	Define a display attribute
.BX	Draw a box around trailing text
.CE	Center trailing text
.CH	Center a heading
.CM	Add a comment
.CP	Copy a help file
.CT	Control help
.DE	Selectively display help text depending on terminal types
.LI	Selectively display help text depending on licensed features
.LN	Draw a line across the screen
.MU	Define a menu line
.NP	Skip to a new page
.OP	Selectively display help text depending on the operating system
.PH	Define a primary heading
.RA	Remove a display attribute
.SH	Define a sub heading
.SP	Skip line(s)
.TI	Define the title line

Each macro is described on a separate page covering the following points, where applicable:

Function
A short description of the function of the macro.

Use
General description of use.

Operands
Description of operands.

Examples
Examples of use.

Notes
Special aspects of use and related information.

See Also
Other related information.

The precise syntax for each macro (that is, how to enter the macro and its operands) is defined in the area within the box towards the top of each page.

The macro itself appears in capital letters at the left hand side of this area. The operands that are applicable to the macro appear to the right of the macro.

Display Attributes

Some of the macros allow characters to be specified that are used as display attributes (for example, to control color and highlighting).

The following predefined attribute characters are provided:

! (hex value X'5A')
This display attribute is used to highlight important help text—for example, the name of a command. It is based on the system variable &ZPOUTHIC.

‘ (hex value X'79')
This display attribute is used for normal help text. It is based on the system variable &ZPLABELC which is the default text display attribute.

| (hex value X'6A')
This display attribute is used for headings when you are not using the .PH and .CH macros. It is based on the system variable &ZPSUBTLC.

Note

This is the EBCDIC broken bar character.

You should use this predefined set of display attributes to ensure that help text is consistent in appearance and conforms to installation standards.

If you need to use one of the above attribute characters as a real character, the .RA help macro can be used to remove a predefined attribute character definition.

Warning

Do not use the asterisk, *****, or ampersand, **&**, characters as display attributes—these characters are reserved by the system.

.AT

Function

Defines a character's display attributes.

.AT *char* [COLOR | COLOUR={BLUE | RED | GREEN | YELLOW | TURQUOISE |PINK | WHITE}] [HLIGHT={USCORE | BLINK | REVERSE | NONE}] [INTENS={HIGH | LOW}] [SUB={YES|NO}

Use

This macro defines a particular character as having certain attributes (color, highlight, intensity) when displayed. The character can then be used to set the appearance of text.

Operands

char

Specifies the character which is to be defined; it can be any printable character.

COLOR | COLOUR={BLUE | RED | GREEN | YELLOW | TURQUOISE | PINK | WHITE}

Specifies the color to be assigned to the character.

HLIGHT={USCORE | BLINK | REVERSE | NONE}

Specifies the type of highlighting to be assigned to the character: underline; blinking; or reverse video.

INTENS={HIGH | LOW}

Specifies the brightness to be assigned to the character.

SUB={YES | NO}

Determines whether variable substitution is to be performed.

SUB=YES should be specified if the help text contains a variable (commencing with an &) that is substituted. This could be used to display system variables, for example.

Examples

```
.AT # COLOR=YELLOW HLIGHT=USCORE
#Hello
```

This example defines the # character to have the display attributes of yellow and underscore. The word `Hello` is displayed in yellow and underlined.

```
.AT # COLOR=PINK HLIGHT=BLINK INTENS=HIGH
#Hello
Goodbye
```

This example displays the word `Hello` in high intensity, blinking, and colored pink. `Goodbye` is shown in green and with low intensity.

Notes

Whenever the character is encountered anywhere in the text after the .AT macro that defined it, the character is replaced by a blank and all text after the character assumes its attributes for the rest of the line (or until another defined attribute character is encountered on the same line).

At the end of each line all attributes are reset to the default values of green color, normal display and low intensity. Therefore if the defined attributes are required on the next line another *char* must be inserted in that line.

Up to 17 different characters can have attributes defined to them. If this limit is exceeded, the remaining .AT macros are ignored and an error message is set pointing to the first macro that exceeded the limit.

See Also

- The .RA macro, which is used to remove a character's attributes.
- The .CT macro, which can be used to redefine the default attributes.

.BX

Function

Draws a box around trailing text.

.BX[*x*][*y*] *text*

Use

This macro draws a box around the text that follows it.

Operands

x

Specifies the character that is to be used for the box border.

y

Specifies a character (or characters) for which display attributes are defined. The box has these attributes.

Note

x and ***y*** can be specified in either order.

text

Specifies the text to be included in the box. The box is sized to fit the text.

The | character (EBCDIC x‘4F’) acts as a line delimiter.

Examples

```
.BX Hello there

This example displays the following default box, in the
default color:

+-----------+
|Hello there|
+-----------+

.AT $ COLOR=BLUE
.AT # COLOR=YELLOW
.BX*$ #Hello there$
```

This example displays the following box, with a blue border of ***** and **Hello there** in yellow:

```
***************
* Hello there *
***************
```

```
.BX | Hello world | this is me ||
```

This example displays a multi-line box, in the default color and border. The **|** character (EBCDIC x‘4F’) acts as a line delimiter.

```
+-------------+
| Hello world |
| this is me  |
+-------------+
```

```
.AT $ COLOR=BLUE
.AT # COLOR=YELLOW
.BX*$ |#Hello there$||
```

This example displays a multi-line box, with a blue border of ***** and **Hello there** in yellow.

```
***************
*             *
* Hello there *
*             *
***************
```

.CE

Function

Centers trailing text.

.CE *text*

Use

This macro centers the text that follows it in the middle of the screen.

Operands

text
Specifies the text to be centered.

Examples

```
.CE Center this text
```

This example displays the words **Center this text** in the middle of the line, when the help panel is displayed.

.CH

Function

Centers a heading.

.CH *text*

Use

This macro centers the text that follows it in the middle of the screen, as a heading.

Operands

text
Specifies the text of the heading to be centered.

Examples

```
.CH Center this heading
```

This example displays the words **Center this heading** in the middle of the help panel when the help text is displayed.

Notes

Text is displayed in the standard heading color and attributes (as set by the &ZPSUBTLC system variable).

.CM

Function

Adds a comment.

.CM *text*

Use

This macro enables you to add a comment to the help text file. Comments are not displayed to the user.

Operands

text
Specifies the text of the comment.

Examples

```
.CM This is a comment.
```

This example does not display the words **This is a comment** on the help panel when the help text is displayed.

Notes

Comments can be placed on any line within the help text.

.CP

Function

Copies a help file.

.CP APPL=*application* [FUNC=*function*] [CROW=*crow*] [CCOL=*ccol*] [FIELD=*field*]

Use

This macro copies the contents of another help text file, and displays the text in place of the .CP macro. The text being copied can be from any level of help.

Operands

APPL=*application*
Specifies the **Application ID** of the application that the help text is to be copied from (3 characters). *This parameter is mandatory.*

FUNC=*function*
Specifies the name of the function that the help text is to be copied from (up to 12 characters).

CROW=*crow*
Specifies the cursor row position of the window that the help text is to be copied from. If this parameter is specified, the ***function*** and ***ccol*** parameters must also be specified.

CCOL=*ccol*
Specifies the cursor column position of the window that the help text is to be copied from. If this parameter is specified, the ***function*** and ***crow*** parameters must also be specified.

FIELD=*field*
Specifies the name of the field that the help text is to be copied from. If this parameter is specified, the ***function*** parameter must also be specified. This parameter is mutually exclusive with the CROW and CCOL parameters.

Examples

```
.CP APPL=$MS FUNC=ADDREC FIELD=FIELD2
```

This example causes the help file for FIELD2 of the application $MS to be imported at display time.

Note

This macro allows you to have a set of common help files that can be copied into other help files at display time. This prevents the duplication of help text, making help easier to maintain.

.CT

Function

Controls the operational characteristics of the display of a help file.

```
.CT [INTEGCHK={YES|NO}]
[CONT={YES|NO}]
[DEFAULT=attrs]
```

Use

Use this macro to tailor the operational characteristics of the display of a help file.

Operands

INTEGCHK

Determines if the help manager checks whether help files that are pointed to by a .MU macro exist when the help is displayed. If **INTEGCHK** is set to YES, and the help file that a .MU macro is pointing to does not exist, the input field for that menu line is replaced by a red asterisk. The default is NO. This is useful when developing help, so that you can see any missing help files that are pointed to by .MU macros.

Note

Checking is not performed for .MU macros that have the CMD parameter specified.
There is a large overhead in having the INTEGCHK option set to YES.

CONT

Determines whether the concatenation of help text lines is performed, if the last non-blank character of a help text line is a plus sign, (+). Valid values are YES and NO; defaults to NO. A continuation should only be used for text lines that contain the .MU macro.

DEFAULT

Redefines the default text display attributes (see the section, *Display Attributes* on page 232). The display attributes are normally the EBCDIC characters **!** (hex value X'5A'—high intensity), **‘** (hex value X'79'—low intensity), and **|** (hex value X'6A'—subtitle highlighting; usually high intensity yellow). You might want to change these default attributes, particularly if the codepoints for these characters are used for alphabetic characters in the code page used on your terminal.

Three characters must be specified: the first replaces the **!** as the high intensity text attribute, the second replaces the **‘** as the low intensity text attribute, and the third replaces the **|** as the subtitle attribute.

Ensure that the .CT DEFAULT= macro is placed before any .RA and .AT macros in the help text.

Examples

```
.CT INTEGCHK=YES CONT=NO
```

This example checks that referenced help files exist and turns off continuation.

```
.CT DEFAULT=<>)
```

In this example **<** replaces **!** as the high intensity default, **>** replaces **‘** as the low intensity default attribute, and **)** replaces **|** as the subtitle color attribute. The characters **!**, **‘**, and **|** appear as themselves.

Note

This example has the same effect as the following six lines:

```
.RA !
.RA ‘
.RA |
.AT < COLOR=&ZPOUTHIC INTENS=HIGH
.AT > COLOR=&ZPLABELC INTENS=LOW
.AT ) COLOR=&ZPSUBTLC INTENS=HIGH
```

See Also

- The .AT macro
- The .MU macro
- The .RA macro

.DE

Function

This macro allows you to mark a section of help text that should be shown only if the region is accessed from an included display terminal type.

```
.DE OPT=START {EXCLUDE|INCLUDE}=(type-1[,type-2[,...]])
.DE OPT=END
```

Use

This macro allows you to specify the environments where you want to display help.

Operands

OPT

Specifies whether you are marking the start or end of a section of text. The valid values are as follows:

START

Specifies that you are marking the start of a section of text that is to be shown only if the region is accessed from an included display terminal type.

END

Specifies that you are marking the end of a section of text.

EXCLUDE|INCLUDE

Determines whether the marked text is shown for the specified display terminal types. The valid values are as follows:

HTML

Specifies that the marked text is for a Web browser.

NONE

Specifies that the marked text is not for any display terminal.

3270

Specifies that the marked text is for a 3270 terminal.

You can either exclude or include certain types. If you exclude certain types, then the types not excluded are included; if you include certain types, then only those types are included.

The list of terminal types must be enclosed in parentheses, ( ), as shown. If you need to specify more than one type, then use commas (,) to separate them, as shown.

When the help manager processes this macro, it determines if the display terminal satisfies the criteria for inclusion. If it is of a valid type, then the text following this macro, up to the associated .DE OPT=END macro, is shown.

Examples

Create a help member with the following text:

```
.DE OPT=START INCLUDE=(HTML)
This help line should be shown on a Web browser
.DE OPT=END
.DE OPT=START EXCLUDE=(HTML)
This help line should be displayed on a 3270 terminal
.DE OPT=END
```

Now view the help from a 3270 terminal by using the V option on the help manager maintenance menus or selection lists. The following is displayed:

```
This help line should be displayed on a 3270 terminal
```

Note

The first line was not displayed, because the 3270 terminal type has not been included.

Notes

Help macros must start in the first column of the help text to be processed.

.LI

Function

Allows you to mark a section of help text that should be displayed only if one or more of the specified features is licensed.

```
.LI OPT=START FEATURES=(feature-1[,feature-2[,…]])
.LI OPT=END
```

Use

This macro enables you to display help that is dependent on the SOLVE product features licensed in a region.

Operands

OPT

Specifies whether you are marking the start or end of a section of text. The valid values are as follows:

START

Specifies that you are marking the start of a section of text that is displayed only if one of the features specified in the FEATURES= parameter is licensed.

END

Specifies that you are marking the end of a section of text.

FEATURES

This parameter specifies one or more feature names. The list of features must be enclosed in parentheses, ( ), as shown. If you need to specify more than one feature, then use commas (,) to separate them, as shown.

When the help manager processes this macro, it determines if at least one of the features specified on this parameter are licensed. If one or more features are licensed, then the text following this macro, up to the associated .LI OPT=END macro, is displayed.

Examples

Create a help member with the following text:

```
.LI OPT=START FEATURES=(licensed_feature)
This help line should be displayed
.LI OPT=END
.LI OPT=START FEATURES=(unlicensed_feature,licensed_feature)
This help line should be displayed too
.LI OPT=END
.LI OPT=START FEATURES=(unlicensed_feature)
This help line should NOT be displayed
.LI OPT=END
```

Note

You need to replace the feature names with the appropriate names, depending on the region you are using.

Now view the help using the V option on the help manager maintenance menus or selection lists. The following is displayed:

```
This help line should be displayed
This help line should be displayed too
```

Note

The last line was not displayed, because the feature was not licensed.

Notes

Help macros must start in the first column of the help text to be processed.

It is possible to have nested .LI macros.

.LN

Function

Draws a line across the screen.

.LN[*x*][*y*]

Use

This macro draws a line across the screen.

Operands

x

Specifies the character which is to be used for the line.

y

Specifies a character (or characters) for which display attributes are defined. The line has these attributes.

Note

x and ***y*** can be specified in either order.

Examples

```
.LN
```

This example draws a line of the default symbol (dashes), in the default color, across the screen.

```
.AT # COLOR=YELLOW
.LN+#
```

This example draws a line of + characters, in yellow, across the screen.

.MU

Function

Defines a menu line.

.MU*nn text* | [APPL=*application*] [FUNC=*function*] [CROW=*crow*] [CCOL=*ccol*] [FIELD=*field*][CMD=*command*]

Use

This macro defines one line of a help menu.

Operands

nn

Specifies the column offset from the left margin to the point where the text is to start.

text

Specifies the text to be displayed on this line of the help menu panel.

|

A delimiter must be placed between the ***text*** and the ***APPL*** parameter.

Note

This is the EBCDIC character X'4F' (solid bar).

APPL=*application*

Specifies the **Application ID** of the application that this menu line's help file is associated with.

FUNC=*function*

Specifies the name of the function that this menu line's help file is associated with.

CROW=*crow*

Specifies the cursor row position of the window that this menu's help file is associated with. If this parameter is specified, the ***function*** and ***ccol*** parameters must also be specified.

CCOL=*ccol*

Specifies the cursor column position of the window that this menu's help file is associated with. If this parameter is specified, the ***function*** and ***crow*** parameters must also be specified.

FIELD=*field*

Specifies the name of the field that this menu's help file is associated with. If this parameter is specified, the ***function*** parameter must also be specified.

CMD=*command*

Specifies the name of a command to be executed if this menu option is selected. This field is mutually exclusive with the APPL, FUNC, CROW and CCOL field parameters.

Note

This command must be defined within the CAS Commands facility.

Examples

```
.CH Sample Menu - Enter S to select

.MU20 Application Level Help|APPL=$SA
.MU20 Function Level Help|APPL=$SA FUNC=SAMPLE
.MU20 Field Based Help|APPL=$SA FUNC=SAMPLE FIELD=SAMPLEFIELD
.MU20 NCS Node Status Display | CMD=$NCSTAT
```

This example displays the help menu shown in Figure 10-14 (in the default heading/text colors and attributes).

Figure 10-14. CAS: Untitled Help

```
SOLVPROD--------------------- CAS : Untitled Help ------------------Page 1 of 1
 Command ===>

                       Sample Menu - Enter S to select

                 _ Application Level Help
                 _ Function Level Help
                 _ Field Level Help
                 _ NCS Node Status Display

              F2=Split     F3=Exit      F4=Return                 F6=HelpHelp
 F7=Backward  F8=Forward   F9=Swap
```

Notes

Each menu line contains a reference to a help file or command. When the help menu is displayed, you can select one or more options by entering **S** in the input

field next to the menu line. You can also place the cursor in a menu line input field and press the ENTER key. The help file associated with the selected menu line is then presented or the command is executed.

When a menu line is defined, the input field for that menu line is automatically displayed.

See Also

The .CT macro.

.NP

Function

Skips to a new page.

```
.NP
```

Use

This macro forces the remainder of the screen to be blank, and displays the rest of the help file on a new page.

Examples

```
This is a page
.NP
This is a new page
```

This example displays the words **This is a page** on the current panel, and **This is a new page** on a new panel.

Notes

You must press the Forward function key or enter the Forward command to view the next page of help.

.OP

Function

Allows you to mark a section of help text that should be displayed only if the region is running on an included operating system.

```
.OP OPT=START {INCLUDE|EXCLUDE}=(system-1[,system-2[,…]])
.OP OPT=END
```

Use

This macro enables you to display help that is specific to the operating system on which a SOLVE region is running.

Operands

OPT

Specifies whether you are marking the start or end of a section of text. The valid values are as follows:

START

Specifies that you are marking the start of a section of text that is to be displayed only if the region is running on an included operating system.

END

Specifies that you are marking the end of a section of text.

INCLUDE|EXCLUDE

Determines whether the marked text is displayed for the specified operating system.

You can either exclude or include certain systems. If you exclude certain systems, then the systems not excluded are included; if you include certain systems, then only those systems are included.

The list of operating systems must be enclosed in parentheses, ( ), as shown. If you need to specify more than one system, then use commas (,) to separate them, as shown.

When the help manager processes this macro, it determines if the current system satisfies the criteria for inclusion. If it is a valid system, then the text following this macro, up to the associated .OP OPT=END macro, is shown.

Examples

Create a help member with the following text:

```
.OP OPT=START INCLUDE=(current-op-sys)
This help line should be displayed
.OP OPT=END
.OP OPT=START INCLUDE=(current-op-sys,another-op-sys)
This help line should be displayed too
.OP OPT=END
.OP OPT=START EXCLUDE=(current-op-sys)
This help line should NOT be displayed
.OP OPT=END
```

Note

You need to replace the operating systems with the appropriate IDs (for example, MVSESA, OS390, and MSPEX). The current ID is displayed on the SOLVE primary menu as the value of OPSYS.

Now view the help using the V option on the help manager maintenance menus or selection lists. The following is displayed:

```
This help line should be displayed
This help line should be displayed too
```

Note

The last line is not displayed, because the current operating system has been excluded.

Notes

Help macros must start in the first column of the help text to be processed.

.PH

Function

Defines a primary heading.

.PH *text*

Use

This macro draws a box around the text that follows it, and displays the text in reverse video.

Operands

text
Specifies the text of the primary heading.

Examples

```
.PH Primary Heading
```

This example displays the following default box border, in the default color, with text in reverse video in the standard heading color (set by the &ZPSUBTLC system variable):

```
+----------------+
|Primary Heading |
+----------------+
```

.RA

Function

Removes a character's display attributes.

.RA *char*

Use

This macro removes a predefined attribute character definition. The character can then be used as a real text character.

Operands

char
Specifies the character which is to have its display attributes removed.

Examples

```
.RA !
This is a test!
```

This example displays the following line on the help panel:

This is a test!

See Also

- The .AT macro, which is used to define a character's attributes.
- The .CT macro, which can be used to redefine all predefined attributes.
- The section titled *Display Attributes*, on page 10-28.

.SH

Function

Defines a sub-heading.

.SH *text*

Use

This macro displays a standard sub-heading line.

Operands

text
Specifies the text of the sub-heading.

Examples

```
.SH This is a sub-heading
```

This example displays the words **This is a sub-heading** in the default sub-heading color (as set by the &ZPSUBTLC system variable).

.SP

Function

Skips lines.

.SP*n*

Use

This macro inserts one or more blank lines in the displayed help file.

Operands

n
Specifies the number of blank lines to insert

Examples

```
Text Line 1
.SP3
Text Line 5
```

This example displays the following:
Text Line 1

Text Line 5

.TI

Function

Defines a title.

.TI[*x*][*y*] *text*

Use

This macro defines a help file's title line.

Operands

x

Specifies the character which is to be used to pad the title line

y

Specifies a character (or characters) for which display attributes are defined. The pad character has these attributes.

Note

x and ***y*** can be specified in either order.

Examples

```
.TI This is a Title
```

This example displays the following title line, using the default pad character (dash) and attributes:

NMID----------This is a Title----------

```
.AT # COLOR=BLUE
.AT % COLOR=YELLOW
.TI# %This is a Title
```

This example displays the following title line, with the pad characters colored blue and the text colored yellow:

NMID----------This is a Title----------

```
.AT # COLOR=BLUE
.AT % COLOR=YELLOW
.TI$% #This is a Title
```

This example displays the following title line, with the pad characters (**$** symbols) colored yellow and the text colored blue:

NMID$$$$$$$$$$$$$$$$$$This is a Title$$$$$$$$$$$$$$$$$$$$$$$$$

Notes

The title line remains in effect until either a new title is defined or the end of the help file is reached.

The value of &NMID is always placed on the left side of the title line.

If no title is defined, a default title line is set up.

11

Maintaining Messages

This chapter describes the Common Application Services (CAS) Message facility.

Messages are items of text which are displayed on a panel in response to specific events (such as error conditions). Use this chapter to define and maintain message definitions.

About Messages

A message definition contains message identifying information as well as the actual message text, an explanation of its purpose and a description of system and user action taken.

Because message definitions are not defined as belonging to particular applications, any defined message can be used by all applications.

To assist in maintaining message definitions, a full-screen text editor is provided.

Defining Messages

You can define a message by selecting the Add Message option from the CAS : Message Definition Menu (or by choosing the ADD action on a list of messages) or by copying an existing message. See *Copying a Message Definition*, on page 11-11.

Message Definition Menu

Messages are accessed through the CAS : Message Definition Menu (option **MS** of the CAS : Maintenance Menu).This panel allows you to add, browse, update, delete, print or copy message definitions, or to obtain a list of messages.

The CAS : Message Definition Menu is illustrated in Figure 11-1:

Figure 11-1. CAS : Message Definition Menu

```
SOLVPROD--------------- CAS : Message Definition Menu ------------------$MS001
Select Option ===>

   A   - Add Message
   B   - Browse Message
   U   - Update Message
   D   - Delete Message
   C   - Copy Message
   L   - List Messages
   P   - Print Message
   V   - View Message
   X   - Exit

Message ID ... ____________ ( Required B U D C P Optional A L )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the CAS : Message Definition Menu are as follows:

Message ID

This message identifier must be entered when adding, browsing, updating, deleting, copying or viewing messages. This identifier cannot contain imbedded blanks and must contain one numeric digit. It cannot contain more than 12 characters.

The following options are available on the CAS : Message Definition Menu:

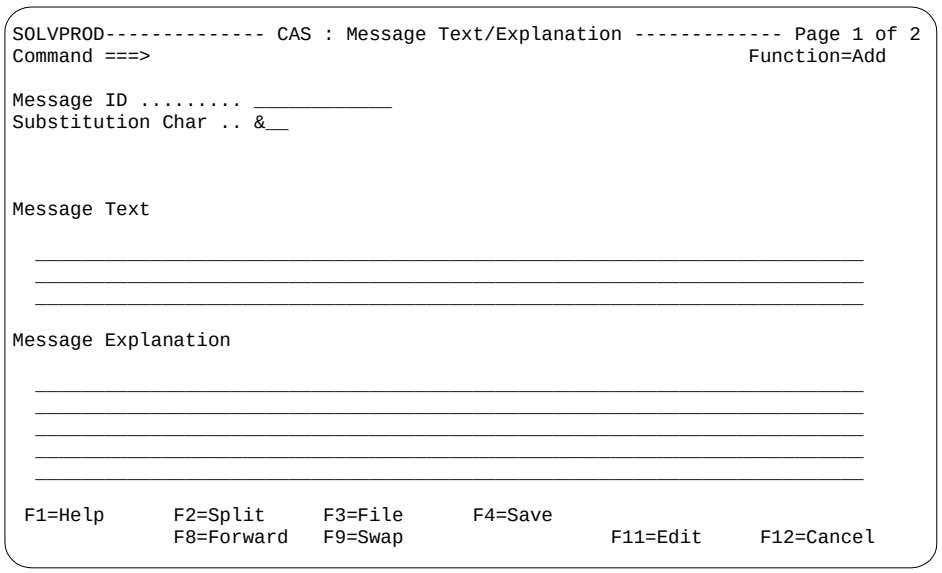
Option	Explanation
A	Add a new message definition
B	Browse a message definition
U	Update a message definition
D	Delete a message definition
C	Copy a message definition
L	List message definitions
P	Print a message definition
V	View a message definition

Detailed descriptions of these options follow.

Adding a Message Definition

Select option **A** to add a new message definition. This displays the CAS : Message Text/Explanation panel, as shown in Figure 11-2.

Figure 11-2. CAS : Message Text/Explanation Panel



The fields displayed on the CAS : Message Text/Explanation Panel as follows:

Message ID

This is the ID that references the message. This identifier cannot contain imbedded blanks and must contain one numeric digit. It cannot contain more than 12 characters. The first two characters usually represent the application that the message belongs to.

Substitution Char

The message text can contain values in variables named P1 - P10 (see Message Text over the page). These variables are prefixed with the Substitution Character which is normally an ampersand (&). However, if you need an (&) to be shown in your message text, you may need to change the Substitution Character to another character - for example, $. In this case the message text might look like this:

THERE ARE $P1 LOCAL & REMOTE SYSTEMS CONNECTED

The Substitution Character also acts as a switch for turning highlighting on or off in Message Explanation text. This also applies to System Action and User Action text on the CAS : Message System/User Action panel.

Message Text

The message text is the text that will be displayed on the SYSMSG line when this message definition is retrieved by an application. This text can contain the variables &P1 through &P10. These variables must be prefixed with the Substitution Character specified above (usually &). An application can provide values to be substituted for the variables &P1 through &P10. Only the variables &P1 through &P10 will be substituted. For example a message text line can contain the following:

```
User &P1 logged on at &P2
```

The application requesting the above message would make the substitutions of the current userid and time for the variables &P1 and &P2, respectively.

Care should be taken when defining messages to be displayed on terminals, to ensure that when any embedded variables are substituted (those prefixed by the Substitution Character), the resulting length does not exceed the terminal screen width.

Message Explanation

The message explanation should state why the message was displayed and explain how the user should interpret the message. For example, the message VSAM ERROR ON FILE &P1 OPT=&P2 FDBK=&P3, should explain what &P1, &P2, and &P3 are, and their possible values.

The first field in the message explanation area is mandatory. The first five lines of message explanation text are displayed on the panel.

If when adding or updating a message, you find that there is more text than can fit on the panel, place the cursor in the message explanation area and press the EDIT function key. This action gives you a full screen text editor that allows you to enter up to 9999 lines of text. Where there is more explanation text than can fit on the panel, a message is displayed that indicates that more explanation text is available.

You can use the Substitution Character to switch highlighting on or off within the message explanation. The switch works on a line by line basis. You must start a new switch sequence on each new line. To highlight words in a line, place the Substitution Character directly before the words to switch highlighting on and directly after the words to switch highlighting off. If it is at the end of a line, the Substitution Character is optional.

If browsing or deleting a message, a message might be displayed indicating that there is more text available than is currently visible. To browse this extra text, place the cursor in the message explanation area and press the Browse function key. This action gives you a full screen editor in browse mode in which you can scroll forward or backward using the appropriate function keys.

For details of the editor commands, refer to Appendix B, *Text Editor Commands.*

After completing this panel, press the FORWARD command key to go to the CAS : Message System/User panel, as shown in Figure 11-3:

Figure 11-3. CAS : Message System/User Action Panel

```
SOLVPROD------------ CAS : Message System/User Action ------------- Page 2 of 2
Command ===>                                                      Function=Add

Message ID ......... _____________
Substitution Char .. &__



System Action

  ____________________________________________________________________________
  ____________________________________________________________________________
  ____________________________________________________________________________
  ____________________________________________________________________________

User Action

  ____________________________________________________________________________
  ____________________________________________________________________________
  ____________________________________________________________________________
  ____________________________________________________________________________

 F1=Help      F2=Split     F3=File      F4=Save
 F7=Backward               F9=Swap                 F11=Edit     F12=Cancel
```

The fields displayed on the CAS : Message System/User Panel are as follows:

Message ID

This is the ID that references the message. This identifier cannot contain imbedded blanks and must contain one numeric digit. It cannot contain more than 12 characters. The first two characters usually represent the application the message belongs to.

Substitution Char

The message text can contain values in variables named P1 - P10 (see Message Text over the page). These variables are prefixed with the Substitution Character which is normally an ampersand (&). However, if you need an & to be shown in your message text, you may need to change the Substitution Character to another character - for example, $. In this case the message text might look like this:

```
THERE ARE $P1 LOCAL & REMOTE SYSTEMS CONNECTED
```

The Substitution Character also acts as a switch for turning highlighting on or off in System Action and User Action text. This also applies to Message Explanation text on the CAS : Message Text/Explanation (previous) panel.

System Action

The system action fields should contain a description of any action taken as a result of the message.

The first field in the system action area is mandatory. The first four lines of system action text are displayed on the panel.

If when adding or updating a message, you find that there is more text than can fit on the panel, place the cursor in the system action area and press the EDIT function key. This action gives you a full screen text editor that allows you to enter up to 9999 lines of text. Where there is more explanation text than can fit on the panel, a message is displayed that indicates that more system action text is available.

You can use the Substitution Character character to switch highlighting on or off with the system action text. The switch works on a line by line basis. You must start a new switch sequence on each new line. To highlight words in a line, place the Substitution Character directly before the words to switch highlighting on and directly after the words to switch highlighting off. If it is at the end of a line, the Substitution Character is optional.

If browsing or deleting a message, a message might be displayed indicating that there is more text available than is currently visible. To browse this extra text, place the cursor in the system action area and press the Browse function key. This action gives you a full screen editor in browse mode in which you can scroll forward or backward using the appropriate function keys.

For details of the editor commands, refer to Appendix B, *Text Editor Commands.*

User Action

The user action fields should contain a description of any action that a user might take as a result of a message.

The first field in the user action area is mandatory. The first four lines of user action text are displayed on the panel.

If when adding or updating a message, you find that there is more text than can fit on the panel, place the cursor in the user action area and press the EDIT function key. This action gives you a full screen text editor that allows you to enter up to 9999 lines of text. Where there is more explanation text than can fit on the panel, a message is displayed that indicates that more user action text is available.

You can use the Substitution Character character to switch highlighting on or off with the user action text. The switch works on a line by line basis. You must start a new switch sequence on each new line. To highlight words in a line, place a Substitution Character directly before the words to switch highlighting on and directly after the words to switch highlighting off. If it is at the end of a line, the Substitution Character is optional.

If browsing or deleting a message, a message might be displayed indicating that there is more text available than is currently visible. To browse this extra text, place the cursor in the user action area and press the Browse function key. This action gives you a full screen editor in browse mode in which you can scroll forward or backward using the appropriate function keys.

After completing the message definition, press the FILE key. To cancel the definition, press the CANCEL key.

Maintaining Message Definitions

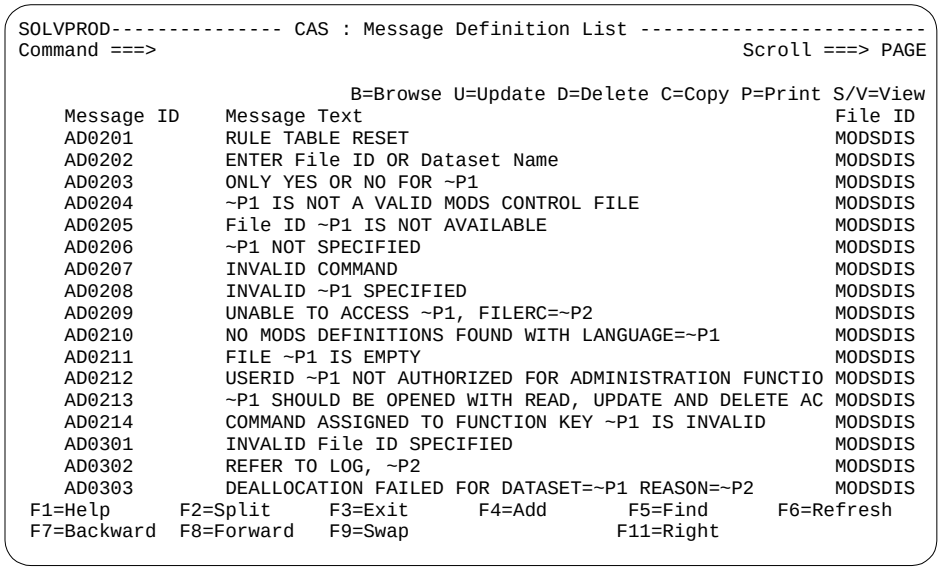
You can browse, update, delete, copy, list, print and view message definitions by selecting the appropriate option from the CAS : Message Definition Menu. You can also perform these functions from a list of message definitions.

Listing Message Definitions

Select option **L** to display the CAS : Message Definition List as shown in Figure 11-4 and Figure 11-5.

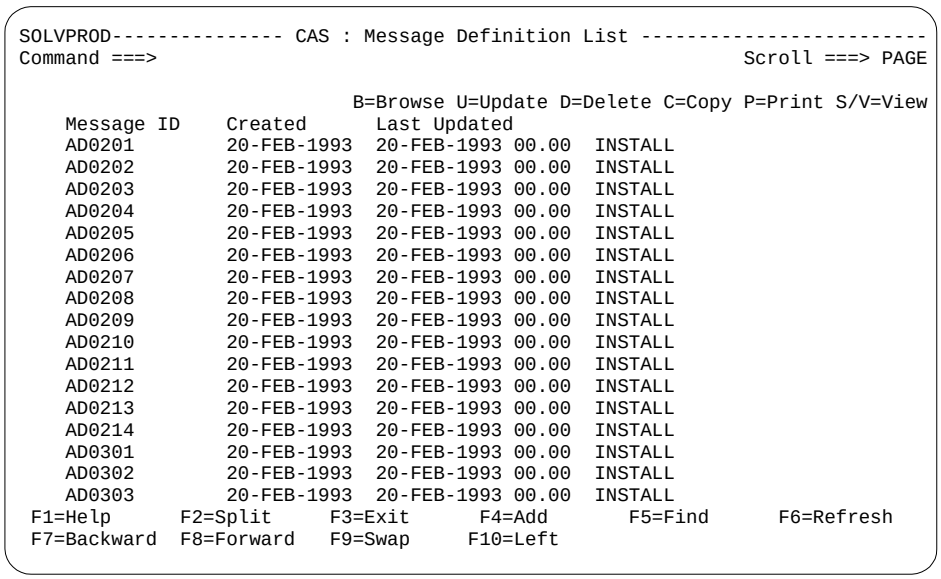
Use the RIGHT and LEFT keys to scroll between the panels.

Figure 11-4. CAS : Message Definition List (page 1)



```
SOLVPROD--------------- CAS : Message Definition List -------------------------
Command ===>                                                  Scroll ===> PAGE

                           B=Browse U=Update D=Delete C=Copy P=Print S/V=View
    Message ID    Message Text                                          File ID
    AD0201        RULE TABLE RESET                                      MODSDIS
    AD0202        ENTER File ID OR Dataset Name                         MODSDIS
    AD0203        ONLY YES OR NO FOR ~P1                                MODSDIS
    AD0204        ~P1 IS NOT A VALID MODS CONTROL FILE                  MODSDIS
    AD0205        File ID ~P1 IS NOT AVAILABLE                          MODSDIS
    AD0206        ~P1 NOT SPECIFIED                                     MODSDIS
    AD0207        INVALID COMMAND                                       MODSDIS
    AD0208        INVALID ~P1 SPECIFIED                                 MODSDIS
    AD0209        UNABLE TO ACCESS ~P1, FILERC=~P2                      MODSDIS
    AD0210        NO MODS DEFINITIONS FOUND WITH LANGUAGE=~P1            MODSDIS
    AD0211        FILE ~P1 IS EMPTY                                     MODSDIS
    AD0212        USERID ~P1 NOT AUTHORIZED FOR ADMINISTRATION FUNCTIO MODSDIS
    AD0213        ~P1 SHOULD BE OPENED WITH READ, UPDATE AND DELETE AC MODSDIS
    AD0214        COMMAND ASSIGNED TO FUNCTION KEY ~P1 IS INVALID       MODSDIS
    AD0301        INVALID File ID SPECIFIED                             MODSDIS
    AD0302        REFER TO LOG, ~P2                                     MODSDIS
    AD0303        DEALLOCATION FAILED FOR DATASET=~P1 REASON=~P2        MODSDIS
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 11-5. CAS : Message Definition List (page 2)



```
SOLVPROD--------------- CAS : Message Definition List -------------------------
Command ===>                                                  Scroll ===> PAGE

                          B=Browse U=Update D=Delete C=Copy P=Print S/V=View
    Message ID    Created      Last Updated
    AD0201        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0202        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0203        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0204        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0205        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0206        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0207        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0208        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0209        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0210        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0211        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0212        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0213        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0214        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0301        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0302        20-FEB-1993  20-FEB-1993 00.00  INSTALL
    AD0303        20-FEB-1993  20-FEB-1993 00.00  INSTALL
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left
```

The fields displayed on the CAS : Message Definition List are as follows:

Message ID

The identifier of the message.

Message Text

The message as it is displayed to the user.

File ID
The identifier of the MODS control file where the message definition is held.

Created
The date that the message was created.

Last Updated
The date and time the message was last updated and the user ID of the person who performed the update.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

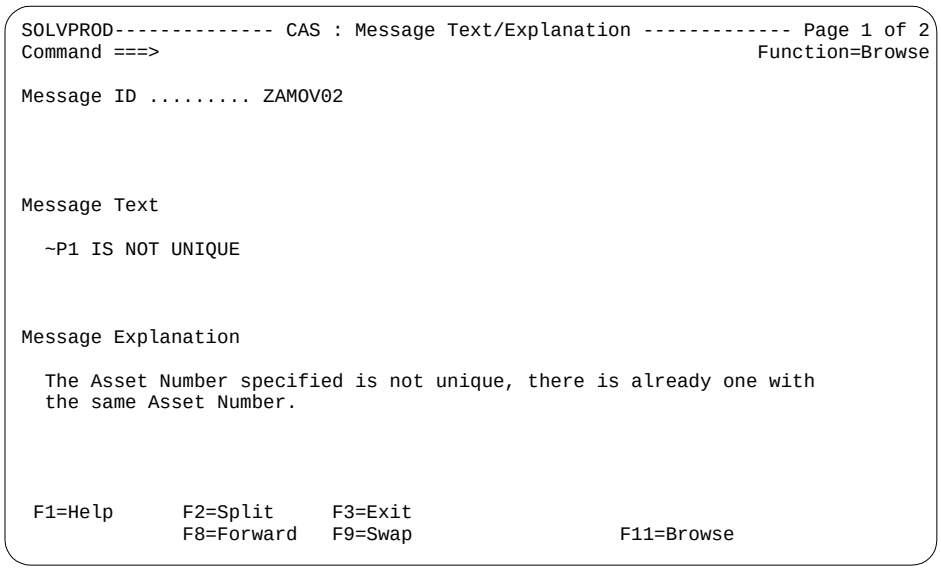
Mnemonic	Action
B	Browse the selected message definition
U	Update the selected message definition
D	Delete the selected message definition
C	Copy the selected message definition
P	Print the selected message definition
V, S, or /	View the selected message definition

These actions operate in the same way as the corresponding menu options.

Browsing a Message Definition

Option **B** displays the Message Text/Explanation panel, as shown in Figure 11-6.

Figure 11-6. CAS : Message Information Panel



You can use the FORWARD and BACKWARD keys to move between panels of a message definition.

To view another message enter its identifier in the **Message ID** field and press ENTER.

Updating a Message Definition

Option **U** displays the CAS : Message Text/Explanation Panel, as shown in Figure 11-2, and the CAS : Message System/User Action Panel (Figure 11-3) except that the function is Update.

After updating the message definition, press the FILE key to save your changes. To cancel the update, press the CANCEL key.

Deleting a Message Definition

Select option **D** to delete a message definition. This displays a message requiring you to confirm the deletion.

Press ENTER to delete the message definition, or press the CANCEL key to cancel the deletion.

Copying a Message Definition

To copy an existing message definition to another (new) message definition, select option **C** from the CAS : Message Definition Menu (Figure 11-1) and provide identification details (***Message ID*** and ***Field Name***) of the message definition to copy from.

Following the above action, the CAS : Message Text/Explanation Panel is displayed, as shown in Figure 11-2, and then the CAS : Message System/User Action panel (Figure 11-3) in Update mode.

Modify these as required. To use the text editor, press the EDIT key, with the cursor positioned in the required input field. For details of the editor commands, refer to Appendix B, *Text Editor Commands.*

After modifying the panel, press the FILE key to create the new message definition. To cancel the copy, press the CANCEL key.

Printing a Message Definition

Select option **P** in order to print a message definition. If you select this option from the menu, you must enter the identifier of the message you want to print in the **Message ID** field on the menu panel.

Viewing a Message Definition

Select option **V** to display a message definition (similar to that shown in Figure 11-2). Use the BACKWARD and FORWARD keys to scroll up and down and use the PREVMSG and NEXTMSG keys to view other message definitions.

12

Maintaining Tables

This chapter describes the Common Application Services (CAS) Table facility. Tables contain sets of table entries that are used to validate data entry.

This chapter describes how to create and maintain table definitions and table entries.

About Tables

A table consists of a table definition and a series of table entries. It defines the location and type of data against which input for a field is validated. Table definitions can be created, updated, browsed, deleted, copied or listed using the CAS : Table Definition Menu (Figure 12-3).

Entries can be stored as data within a MODS control file. The entries can also be **INFO/MASTER** field names, values of an **INFO/MASTER** field, or Object Services attributes.

A table contains one entry for each valid value of the input field. Table entries can be created, updated, browsed, deleted, copied or listed through the CAS : Table Entry Menu (Figure 12-2 on page 12-4).

Note

Table entries can only be added to tables that have an **Edit type** of **TABLE** (see **Edit type**, in the section titled *Adding a Table Definition*, on page 12-6).

Reloading Validation Tables

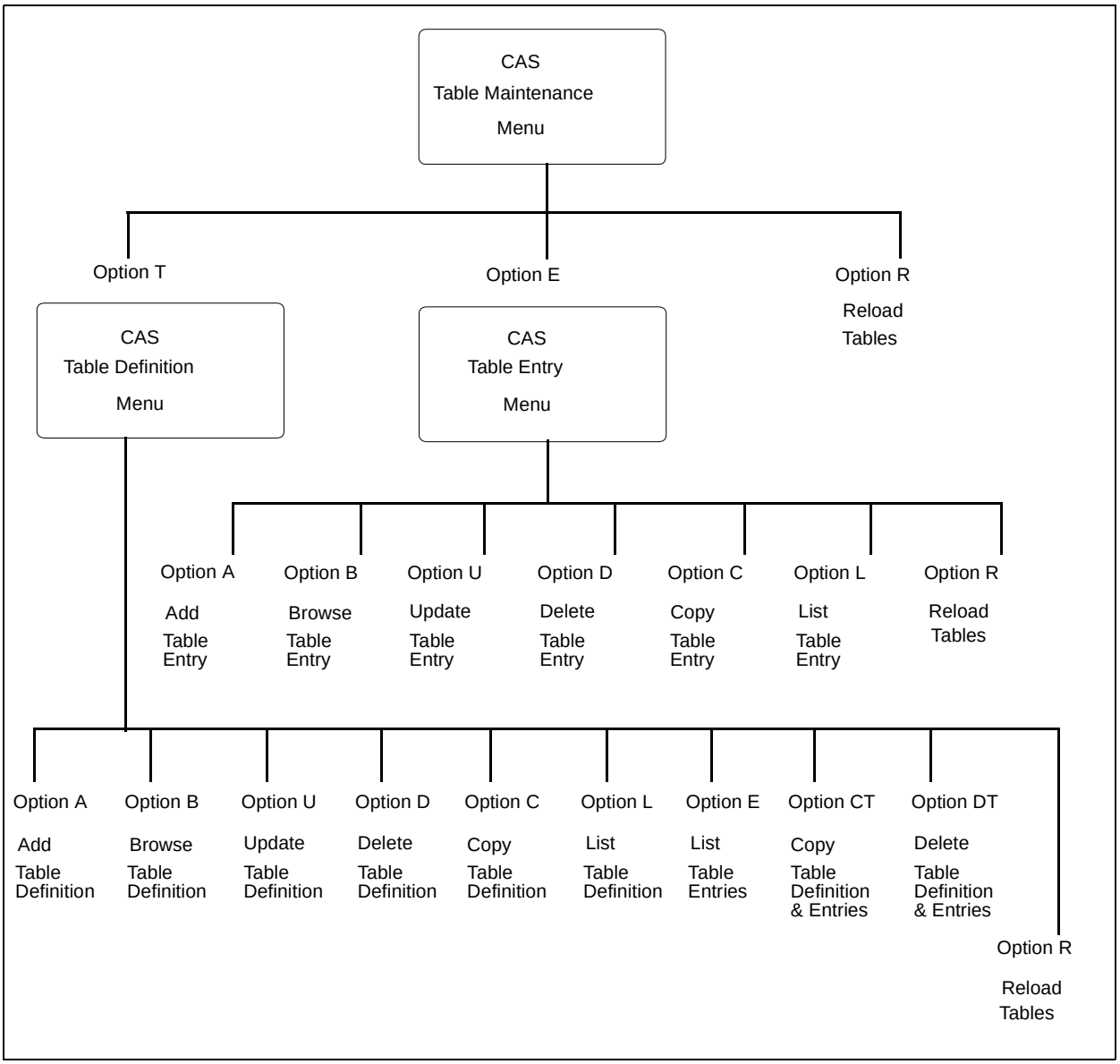
Whenever table entries or table definitions are added, updated or deleted (from a table that has already been loaded), a reload must be performed. This means that all table definitions for the application are loaded from the control file into memory.

If a table is flagged for loading and has an **Edit type** of TABLE, all its entries are also loaded into memory.

As illustrated in Figure 12-1, a reload can be initiated from either the CAS : Table Maintenance Menu, the CAS : Table Definition Menu or the CAS : Table Entry Menu.

Enter the **Application ID** of the application whose tables require reloading, and select option **R**. A confirmation message is displayed when the reload is complete.

Figure 12-1. CAS Table Maintenance Panel Navigation



Defining and Maintaining Table Definitions

This section describes how to create and maintain table definitions.

Table Maintenance Menu

Tables are accessed through the CAS : Table Maintenance Menu (option **T** of the CAS : Maintenance Menu), as illustrated in Figure 12-2:

Figure 12-2. CAS : Table Maintenance Menu

```
SOLVPROD----------------- CAS : Table Maintenance Menu -------------------$VM002
 Select Option ===>

    T   - Table Definitions
    E   - Table Entries
    R   - Reload Tables
    X   - Exit

 Appl ID .........+ ___ ( Required R  Optional T E )









  F1=Help      F2=Split     F3=Exit      F4=Return
                            F9=Swap
```

The input field displayed on the CAS : Table Maintenance Menu is as follows:

Application ID

The 3 character identifier of the application to which the validation table(s) belong. It must be entered when reloading validation tables. When moving to the Table Definition Menu or the Table Entry Menu, the **Application ID** can either be entered here or on the next panel. Prompt support is available for this field.

The following options are available on the CAS : Table Maintenance Menu:

Option	Explanation
T	Go to the **Table Definition Menu**
E	Go to the **Table Entry Menu**
R	Reload an application's tables

Validation tables can be defined, browsed, updated, copied, deleted, or listed through the Table Definition Menu (option **T** of the CAS : Table Maintenance Menu), as illustrated in Figure 12-3.

Figure 12-3. CAS : Table Definition Menu

```
SOLVPROD---------------- CAS : Table Definition Menu --------------------$VM011
Select Option ===>

   A   - Add Table
   B   - Browse Table
   U   - Update Table
   D   - Delete Table
   C   - Copy Table
   L   - List Tables
   E   - List Table Entries
   CT  - Copy Table and Table Entries
   DT  - Delete Table and Table Entries
   R   - Reload Tables
   X   - Exit

Appl ID .........+ ___          ( Required B U D C L E CT DT R )
Field name ....... ____________ ( Required B U D C E CT DT Optional R )




 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The input fields displayed on the CAS : Table Definition Menu are as follows:

Application ID

The 3-character identifier of the application to which the table belongs. It must be entered for all options except Add Table.

Field name

This (up to) 12-character name of the field with which the table is associated, must be entered when browsing, updating, deleting, or copying validation tables. When obtaining a list of table definitions, the **Field name** field is optional. If it is entered, (either partially or fully), table definitions that generically match the given value are listed. If it is left blank, all table definitions belonging to the application will be included. When adding a table definition, the **Field name** field can either be entered here or on the next panel.

If the field name is entered when reloading an application's tables, only the nominated table is reloaded. If it is left blank, all tables for the nominated application are reloaded.

The following options are available from the CAS : Table Definition Menu:

Option	Explanation
A	Add a new table definition
B	Browse an existing table definition
U	Update an existing table definition
D	Delete a table definition
C	Copy an existing table definition
L	List table definitions
E	List the entries in a table
CT	Copy an existing table definition, plus all its table entries
DT	Delete a table definition and all its table entries
R	Reload an application's tables

Detailed descriptions of these options follow.

Adding a Table Definition

Defining a table requires you to fill in two panels. The first of these defines the actual table, while the second specifies all the fields required for each entry in the table.

Option **A** initially displays the first page of the Table Definition panel, as shown in Figure 12-4.

Figure 12-4. CAS : Table Description Panel

```
SOLVPROD------------------- CAS : Table Description -----------------Page 1 of 2
 Command ===>                                                    Function=Add

 Appl ID .................. ___
 Field name ............... ____________
 Field description ........ ____________________
 Edit type ................ TABLE (TABLE, OSATT, OSFLD, IMFLD or IMREC)
 For Edit type = TABLE:
    Validation exit ................. ________
    Sequence numbers ................ ___ (YES or NO)
    Load table? ..................... ___ (YES or NO)
    Max abbreviation length ......... ___ (3 - 8 or blank if none)
    Max full value length ........... ___ (3 - 20)
    Max description length .......... ___ (3 - 38 or blank if none)
 For Edit type = IMFLD or IMREC:
    INFO/MASTER category ............ ____
 For Edit type = IMREC:
    INFO/MASTER field ............... ____________
    INFO/MASTER description field ... ____________
 For Edit type = OSATT or OSFLD:
    Object Services Class ID ........ ____________

  F1=Help      F2=Split     F3=File      F4=Save
               F8=Forward   F9=Swap                          F12=Cancel
```

The input fields displayed on the CAS : Table Description Panel are as follows:

Application ID
The 3-character identifier of the application to which the table belongs. The application ID must already be defined in the Application Register.

Field name
The (up to) 12-character name of the field with which the table is associated.

Field description
This 1- to 20-character description of the field is displayed to the user, in error messages and help.

Edit type
Specifies how valid values for the field are stored. The **Edit type** must be one of the following:

TABLE
Valid values are stored as table entries within the CAS control file.

IMFLD
Valid values are the field names defined in the specified category of an **INFO/MASTER** database.

IMREC
Valid values are the data values for a field in the specified category of an **INFO/MASTER** database.

OSATT
Valid values are stored as attribute IDs within the specified class of Object Services.

OSFLD
Valid values are the database field names of the attributes within the specified class of Object Services.

Validation exit
Specifies (optionally) an NCL procedure to validate data entered on the Table Entry Panel (Figure 12-13). It is called when table entries are added, updated or deleted.

Sequence numbers?
Allows you to control the order in which valid values are presented when a user requests assistance (by entering a question mark in an input field).

- If you enter **NO**, the values are presented in the standard order (sorted alphabetically).
- If you enter **YES**, you are prompted for a sequence number when defining each table entry. The values are presented in the order specified.

This field is mandatory when the **Edit Type** is **TABLE.**

Load table?

If **YES** is entered here, all entries in the table are loaded into memory for increased performance. This is recommended for all tables that are not unduly large.

This field is mandatory when the **Edit Type** is **TABLE.**

Max abbreviation length

Specifies the maximum length (in the range 3 to 8) of the **Abbreviated value** field on the Table Entry Panel (Figure 12-13). If no abbreviated values are required, leave this field blank.

Max full value length

Specifies the maximum length of the **Full value** field on the Table Entry Panel. Enter a number between 3 and 20.

This field is mandatory when the **Edit Type** is **TABLE.**

Max description length

Specifies the maximum length of the **Description** field on the Table Entry Panel. Enter a number between 3 and 38. If no descriptions of entries are required, leave this field blank.

INFO/MASTER category

The name of the INFO/MASTER category which is to be searched for matching field values. Required only when the **Edit Type** of the table is **IMREC** or **IMFLD**.

INFO/MASTER field

The name of the **INFO/MASTER** field - within the specified category—whose defined values comprise the list of valid values. Required only when the **Edit Type** of the table is **IMREC**.

INFO/MASTER description field

The name of the field in the INFO/MASTER records which contains the description of the value of the **INFO/MASTER** field in the same record. Applicable only when the **Edit type** of the table is **IMREC**. This field is optional.

Object Services Class ID

The ID of the Object Services Class whose defined attributes or database field names comprise the list of valid values. Required only when the **Edit Type** of the table is **OSATT** or **OSFLD**.

If the table has an **Edit type** of **TABLE**, you have the option of specifying additional input fields to appear on the Table Entry panel (Figure 12-13):

- If you do not require any extra data for the table entries, press the FILE key to create the table definition. To cancel the create, press the CANCEL key.
- To define extra input fields for the Table Entry panel, press the FORWARD key, to go to the CAS : Table Text Fields panel (as illustrated in Figure 12-5).

Figure 12-5. CAS : Table Text Fields Panel

```
SOLVPROD------------------ CAS : Table Text Fields -----------------Page 2 of 2
Command ===>                                                    Function=Update

Appl ID .................. TST
Field name ............... TST001


                  Field   Text description        Length(4-38)

                    1     I/M Application_____    4___
                    2     Service Procedure___    8___
                    3     System Name_________    24__
                    4     Record Category_____    24__
                    5     ____________________    ____
                    6     ____________________    ____
                    7     ____________________    ____
                    8     ____________________    ____
                    9     ____________________    ____
                   10     ____________________    ____



 F1=Help      F2=Split     F3=File      F4=Save                  F6=Entries
 F7=Backward               F9=Swap                              F12=Cancel
```

Enter the following two fields once for each data input field that you require on the Table Entry Panel (see Figure 12-13):

Text Description

This 1 to 20 character description of the input field is displayed as an input prompt on the Table Entry Panel.

Length

The maximum length to which data can be entered in the input field. A numeric value between 4 and 38 is required. Each **Text Description** must have a **Length** specified.

After defining the validation table, press the FILE key. To cancel the table specification, press the CANCEL key.

Browsing a Table Definition

Option B displays the table definition, as shown in Figure 12-4 and Figure 12-5.

If the edit type is TABLE, you can press the ENTRIES key to display the Table Entry Definition List (Figure 12-14).

Updating a Table Definition

Option **U** displays the Table Description panel (as illustrated in Figure 12-4) and the Table Text Fields Panel (Figure 12-5). Refer to the section titled *Adding a Table Definition*, on page 12-6 for further details.

After updating the table definition, press the FILE key. To cancel the update, press the CANCEL key.

If the edit type is TABLE, you can press the ENTRIES key to display the Table Entry Definition List (Figure 12-14). Edit the entries as required.

Deleting a Table Definition

Select option **D** to delete a table definition. This displays the Table Description panel (Figure 12-4) which you use to confirm the deletion. Press ENTER to delete the table definition, or press the CANCEL key to cancel the deletion.

If there are table entries associated with the table definition, the delete operation is rejected. Use option **DT** to delete a table definition and its entries.

See *Deleting a Table Definition and Its Table Entries*, on page 12-15.

Copying a Table Definition

To copy an existing table definition to another (new) table definition, select option **C** from the Table Definition Maintenance Menu, and provide the **Application ID** and **Field name** of the table definition to copy from.

This option displays the Table Description panel (Figure 12-4) and the Table Text Fields panel (Figure 12-5) of the new table definition. Refer to the section titled *Adding a Table Definition*, on page 12-6 for further details.

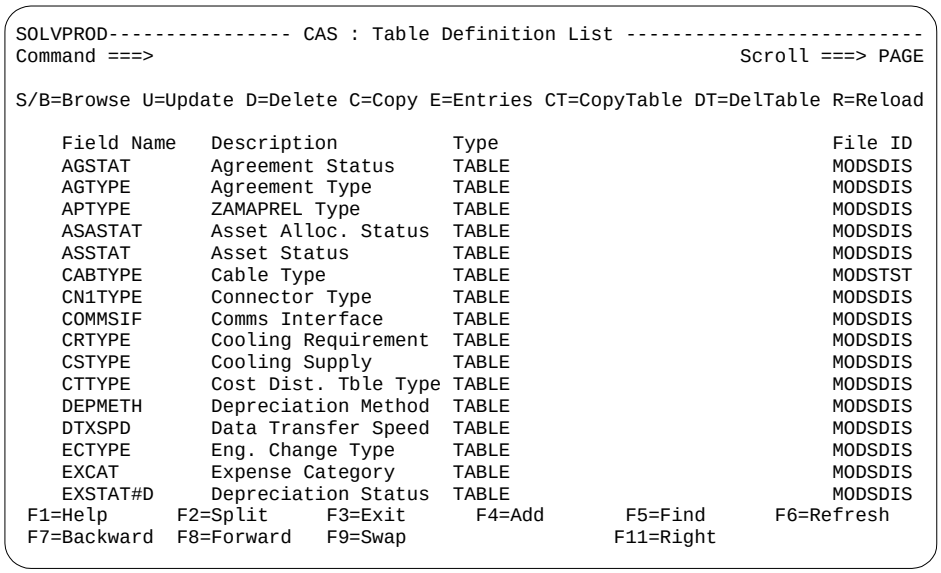
After entering the new table definition, press the FILE key. To cancel the copy, press the CANCEL key.

This operation copies only the table definition, not its entries. In order to copy the table entries as well use option **CT**. See *Copying a Table Definition and Its Table Entries*, on page 12-15.

Listing Table Definitions

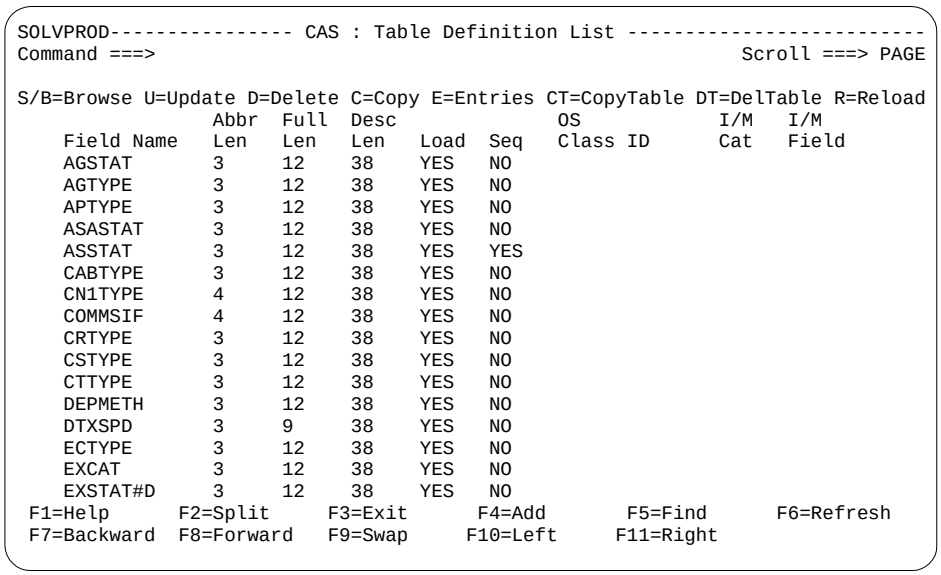
Option **L** displays the CAS : Table Definition List as shown in Figure 12-6, Figure 12-7, and Figure 12-8. Use the RIGHT and LEFT keys, to scroll between the panels.

Figure 12-6. CAS : Table Definition List (page 1)



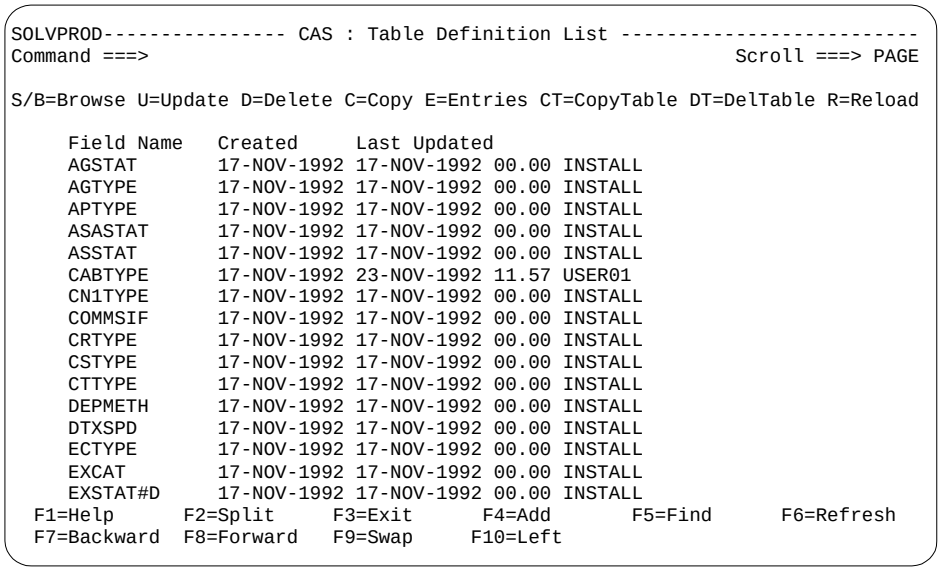
```
SOLVPROD---------------- CAS : Table Definition List -------------------------
Command ===>                                                 Scroll ===> PAGE

S/B=Browse U=Update D=Delete C=Copy E=Entries CT=CopyTable DT=DelTable R=Reload

    Field Name   Description          Type                              File ID
    AGSTAT       Agreement Status     TABLE                             MODSDIS
    AGTYPE       Agreement Type       TABLE                             MODSDIS
    APTYPE       ZAMAPREL Type        TABLE                             MODSDIS
    ASASTAT      Asset Alloc. Status  TABLE                             MODSDIS
    ASSTAT       Asset Status         TABLE                             MODSDIS
    CABTYPE      Cable Type           TABLE                             MODSTST
    CN1TYPE      Connector Type       TABLE                             MODSDIS
    COMMSIF      Comms Interface      TABLE                             MODSDIS
    CRTYPE       Cooling Requirement  TABLE                             MODSDIS
    CSTYPE       Cooling Supply       TABLE                             MODSDIS
    CTTYPE       Cost Dist. Tble Type TABLE                             MODSDIS
    DEPMETH      Depreciation Method  TABLE                             MODSDIS
    DTXSPD       Data Transfer Speed  TABLE                             MODSDIS
    ECTYPE       Eng. Change Type     TABLE                             MODSDIS
    EXCAT        Expense Category     TABLE                             MODSDIS
    EXSTAT#D     Depreciation Status  TABLE                             MODSDIS
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 12-7. CAS : Table Definition List (page 2)



```
SOLVPROD---------------- CAS : Table Definition List -------------------------
Command ===>                                                 Scroll ===> PAGE

S/B=Browse U=Update D=Delete C=Copy E=Entries CT=CopyTable DT=DelTable R=Reload
                Abbr  Full  Desc             OS            I/M   I/M
    Field Name  Len   Len   Len   Load  Seq  Class ID      Cat   Field
    AGSTAT      3     12    38    YES   NO
    AGTYPE      3     12    38    YES   NO
    APTYPE      3     12    38    YES   NO
    ASASTAT     3     12    38    YES   NO
    ASSTAT      3     12    38    YES   YES
    CABTYPE     3     12    38    YES   NO
    CN1TYPE     4     12    38    YES   NO
    COMMSIF     4     12    38    YES   NO
    CRTYPE      3     12    38    YES   NO
    CSTYPE      3     12    38    YES   NO
    CTTYPE      3     12    38    YES   NO
    DEPMETH     3     12    38    YES   NO
    DTXSPD      3     9     38    YES   NO
    ECTYPE      3     12    38    YES   NO
    EXCAT       3     12    38    YES   NO
    EXSTAT#D    3     12    38    YES   NO
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 12-8. CAS : Table Definition List (page 3)



```
SOLVPROD---------------- CAS : Table Definition List -------------------------
Command ===>                                                  Scroll ===> PAGE

S/B=Browse U=Update D=Delete C=Copy E=Entries CT=CopyTable DT=DelTable R=Reload

     Field Name   Created     Last Updated
     AGSTAT       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     AGTYPE       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     APTYPE       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     ASASTAT      17-NOV-1992 17-NOV-1992 00.00 INSTALL
     ASSTAT       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     CABTYPE      17-NOV-1992 23-NOV-1992 11.57 USER01
     CN1TYPE      17-NOV-1992 17-NOV-1992 00.00 INSTALL
     COMMSIF      17-NOV-1992 17-NOV-1992 00.00 INSTALL
     CRTYPE       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     CSTYPE       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     CTTYPE       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     DEPMETH      17-NOV-1992 17-NOV-1992 00.00 INSTALL
     DTXSPD       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     ECTYPE       17-NOV-1992 17-NOV-1992 00.00 INSTALL
     EXCAT        17-NOV-1992 17-NOV-1992 00.00 INSTALL
     EXSTAT#D     17-NOV-1992 17-NOV-1992 00.00 INSTALL
  F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
  F7=Backward  F8=Forward   F9=Swap     F10=Left
```

The fields displayed on the panels of the Table Definition List are as follows:

Field Name
The identifier of the field to which the table applies.

Description
A description of the purpose of the table.

Type
The type of the table; TABLE, IMFLD, IMREC, OSATT, or OSFLD.

File ID
The identifier of the highest level MODS control file in which the definition is present.

Abbr Len
The maximum length of the abbreviations of valid values in the table.

Full Len
The maximum length of values in the table.

Desc Len
The maximum length of the description of entries in the table.

Load
Whether the table is loaded into memory.

Seq
Whether the values are displayed in sequence order.

OS Class ID
The identifier of the Object Services class from which entries are drawn.

I/M Cat
The identifier of the INFO/MASTER category from which entries are drawn.

I/M Field
The identifier of the **INFO/MASTER** field from which entries are drawn.

Created
The date the table definition was created.

Last Updated
The date and time that the table definition was last updated. The user ID of the person who updated the table definition is also displayed. If the definition has not been updated since installation, INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
B, S, or /	Browse the selected table definition
U	Update the selected table definition
D	Delete the selected table definition
C	Copy the selected table definition
E	List the table's entries
CT	Copy an existing table definition, plus all its table entries
DT	Delete a table definition and all its table entries
R	Reload the table into memory

Listing Table Entries

Option E displays the CAS : Table Entry Definition List as shown in Figure 12-9, Figure 12-10, and Figure 12-11. Use the RIGHT and LEFT keys to scroll between the panels.

Figure 12-9. CAS : Table Entry Definition List (page 1)

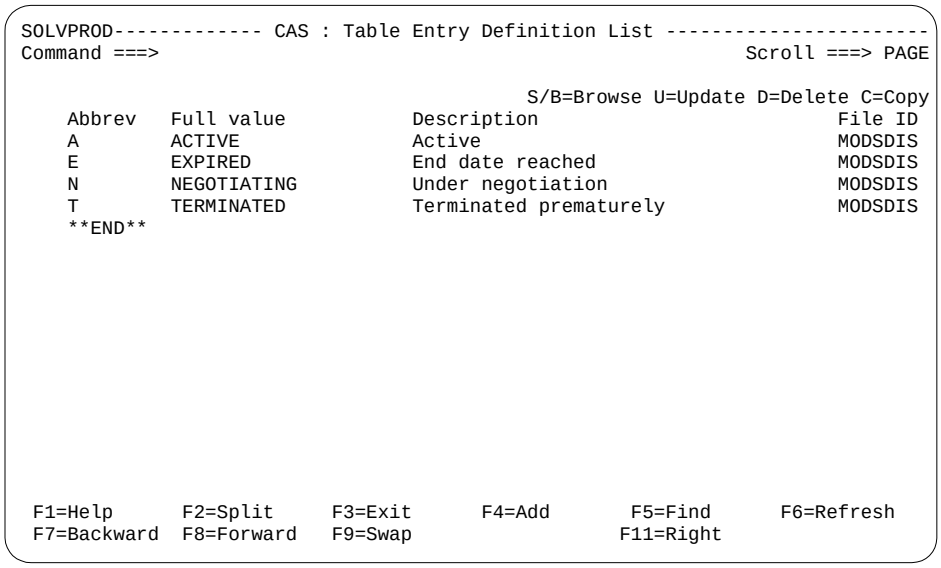


Figure 12-10. CAS : Table Entry Definition List (page 2)

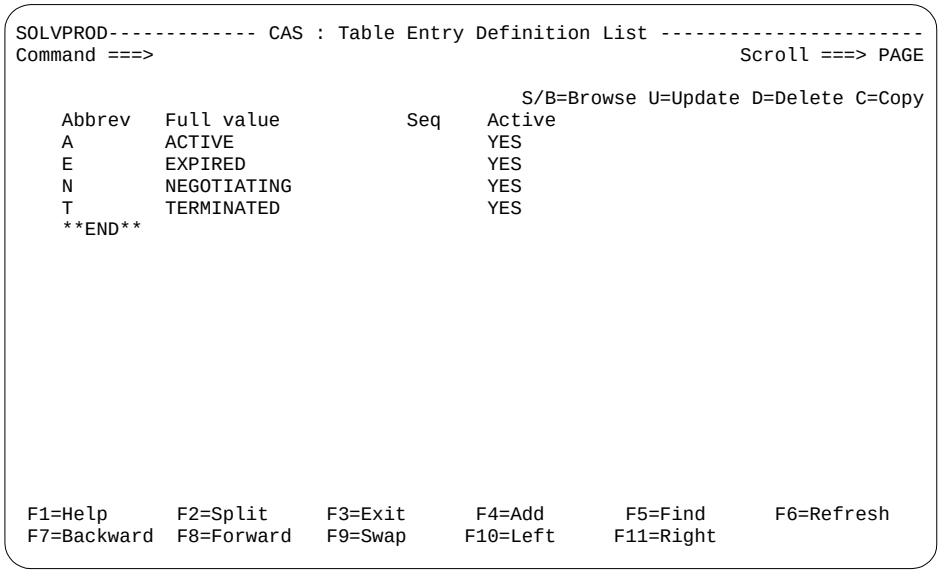
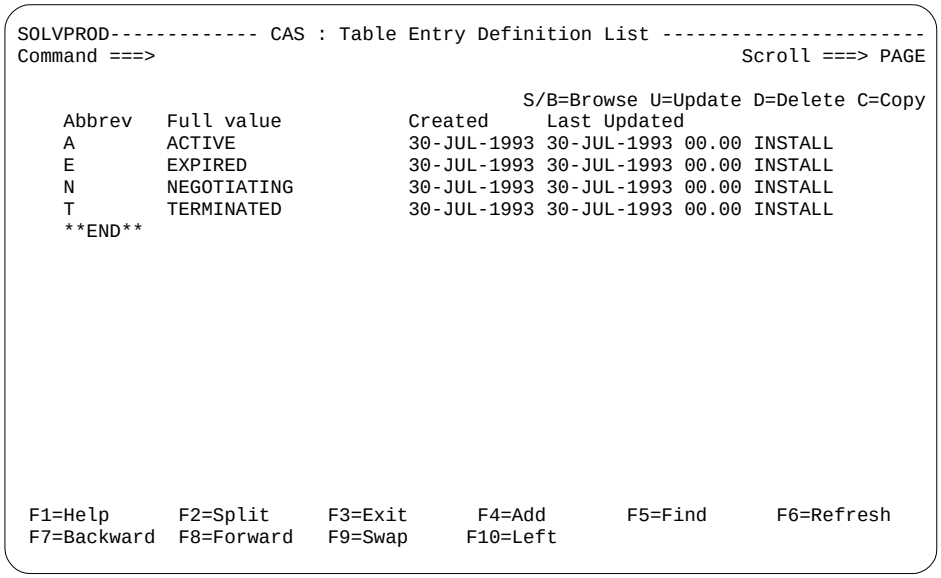


Figure 12-11. CAS : Table Entry Definition List (page 3)



Copying a Table Definition and Its Table Entries

To copy an existing table definition, plus its entries, to another (new) table definition, select option **CT** from the Table Definition Menu and enter the **Application ID** and **Field Name** fields to specify the table definition to copy from.

After completing the above actions, the Table Description panel (as illustrated in Figure 12-4) and the Table Text Fields panel (Figure 12-5) of the new table definition are displayed. Refer to the section titled *Adding a Table Definition*, on page 12-6 for further details.

When you have finished the new table definition, press the FILE key. To cancel the copy, press the CANCEL key.

Deleting a Table Definition and Its Table Entries

To delete a table definition plus all entries in the table, select option **DT**.

This option displays the Table Description panel (Figure 12-4). This panel is displayed so that you can confirm the deletion. Press ENTER to delete the validation table definition and entries, or press the CANCEL key to cancel the deletion.

Maintaining Table Entries

Table entries represent the valid values for a data entry field. You must define a table definition before you can define its table entries.

Table Entry Definition Menu

Table entries can be added, browsed, updated, copied, deleted, or listed through the Table Entry Menu (option **E** of the CAS : Table Maintenance Menu), as illustrated in Figure 12-12.

Figure 12-12. CAS : Table Entry Definition Menu

```
SOLVPROD------------ CAS : Table Entry Definition Menu ----------------$VM021
Select Option ===>

   A   - Add Table Entry
   B   - Browse Table Entry
   U   - Update Table Entry
   D   - Delete Table Entry
   C   - Copy Table Entry
   L   - List Table Entries
   R   - Reload Tables
   X   - Exit

Appl ID .........+ ___          ( Required all )
Field Name ......+ ____________ ( Required A B U D C L Optional R )
Full Value ....... ____________ ( Required B U D C )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The input fields displayed on the Table Entry Definition Menu are as follows:

Application ID

The 3-character identifier of the application to which the table belongs. It must be entered for all options.

Field Name

The (up to) 12-character name of the field with which the table is associated. This name must be entered when adding, browsing, updating, deleting, copying or listing validation table entries.

If the field name is entered when reloading an application's tables, only the nominated table is reloaded. If it is left blank, all tables for the nominated application are reloaded.

Full Value

The full value for the table entry. You must make an entry in this field when browsing, updating, deleting, or copying a record.

You can enter a partial value in this field when using the List option. This restricts the list to records that match the partial value you entered.

The following options are available on the Table Entry Menu:

Option	Explanation
A	Add a new table entry
B	Browse an existing table entry
U	Update an existing table entry
D	Delete a table entry
C	Copy an existing table entry
L	List table entries
R	Reload an application's tables

Detailed descriptions of these options follow.

Adding a Table Entry

Option **A** displays the CAS : Table Entry Definition panel, as shown in Figure 12-13.

Figure 12-13. CAS : Table Entry Definition Panel

```
SOLVPROD--------------- CAS : Table Entry Definition ---------------Page 1 of 1
Command ===>                                                      Function=Add

Appl ID .................. TST
Field name ............... TST001

Full value ............... ____________________
Abbreviated value ........ ___
Description .............. ____________________________
Sequence number .......... ______
Active? .................. YES (YES or NO)

I/M Application .......... ____
Service Procedure ........ _________
System Name .............. ____________________________
Record Category .......... ____________________________




 F1=Help      F2=Split     F3=File      F4=Save
                           F9=Swap                              F12=Cancel
```

Note

Only **Appl ID**, **Field name**, **Full value,** and **Active?** fields appear on every Table Entry Definition panel. The other fields depend on how the table is defined.

The input fields displayed on the Table Entry Definition panel are as follows:

Appl ID

The 3-character identifier of the application to which the table belongs

Field Name

The (up to) 12-character name of the field with which the table is associated.

Full Value

The full value of the table entry.

Abbreviated Value

Specifies an abbreviated value which can be entered in an input field by the user. The abbreviation is validated and replaced by the corresponding **Full Value**.

Note

Abbreviated Value does not appear on the Table Entry Panel unless a **Max** *abbreviation length* was specified in the table definition.

Sequence Number

A 6-digit number that specifies the order in which valid values for a field (that is, table entries) are displayed on a selection list.

Note

This field is not displayed unless the **Sequence numbers** field was set to YES in the table definition.

Description

A description of the table entry.

Note

The **Description** field does not appear on the Table Entry panel unless a **Max description length** was specified in the table definition.

Active

Flags the table entry as being either currently valid (YES) or obsolete (NO).

Validation is performed using both active and inactive table entries. You can allow both inactive and active entries to be entered on a search panel but only active values when adding a record. See OPT=VALIDATE in 18, *CAS Programming Interface ($CACALL)* for further details.

Any additional fields that are specified in the table definition (see Figure 12-5, the Table Text Fields panel) also appear on the Table Entry Definition panel. Complete these fields as required.

After specifying the table entry, press the FILE key. To cancel the add, press the CANCEL key.

Browsing a Table Entry

Select option **B** to display the Table Entry Definition panel, as shown in Figure 12-13.

Use the scroll keys, FORWARD and BACKWARD, to move between the panels of the definition.

Updating a Table Entry

Select option **U** to display the Table Entry Definition panel (as illustrated in Figure 12-13). Refer to the section *Adding a Table Entry*, on page 12-17 for further details.

After updating the table entry, press the FILE key. To cancel the update, press the CANCEL key.

Deleting a Table Entry

To delete a table entry, select option **D**. The Table Entry Definition panel is displayed allowing you to confirm the deletion.

Press ENTER to delete the table entry, or press the CANCEL key to cancel the deletion.

Note

A table entry that is no longer valid should not be deleted, as old records can still contain the value. Instead, set the ***Active*** flag to NO.

Copying a Table Entry

To copy an existing table entry to another (new) table entry, select option **C** from the Table Entry Definition Menu, and specify the entry (using the menu fields) that you want to copy.

The above action displays the Table Entry Definition panel (see Figure 12-13) of the new table entry. Modify this panel as required. Refer to the section titled *Adding a Table Entry*, on page 12-17 for further details.

After entering the new table entry, press the FILE key. To cancel the copy, press the CANCEL key.

Listing Table Entries

Option **L** displays the CAS : Table Entry Definition List, as shown in Figure 12-14, Figure 12-15 and Figure 12-16.

Use the RIGHT and LEFT keys to scroll between the panels.

Figure 12-14. CAS : Table Entry Definition List (page 1)

```
SOLVPROD-------------- CAS : Table Entry Definition List -----------------------
 Command ===>                                                 Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
     Abbrev   Full value           Description                          File ID
     X20      X20 CONTROLLER       Controller for X20s                  MODSUSR
     **END**
```

```
  F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
  F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 12-15. CAS : Table Entry Definition List (page 2)

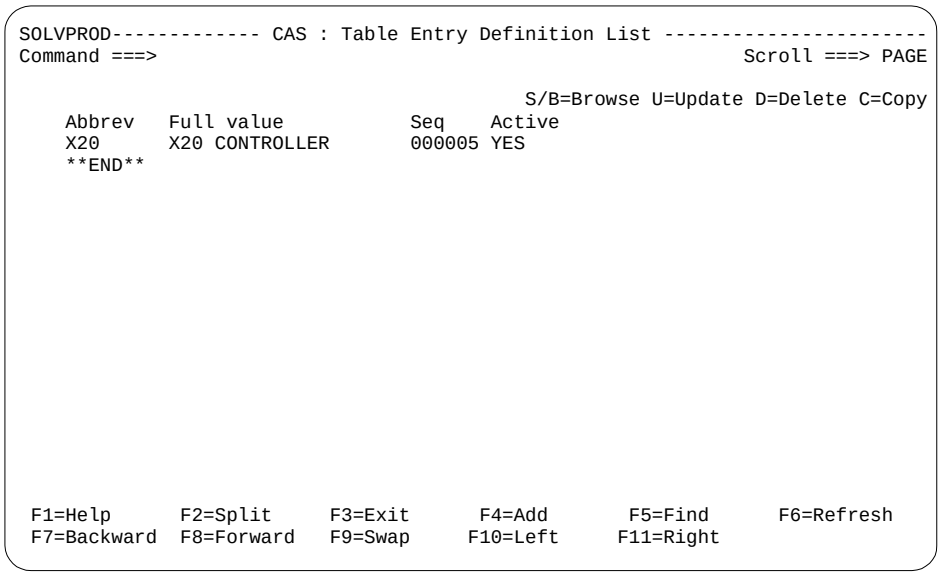
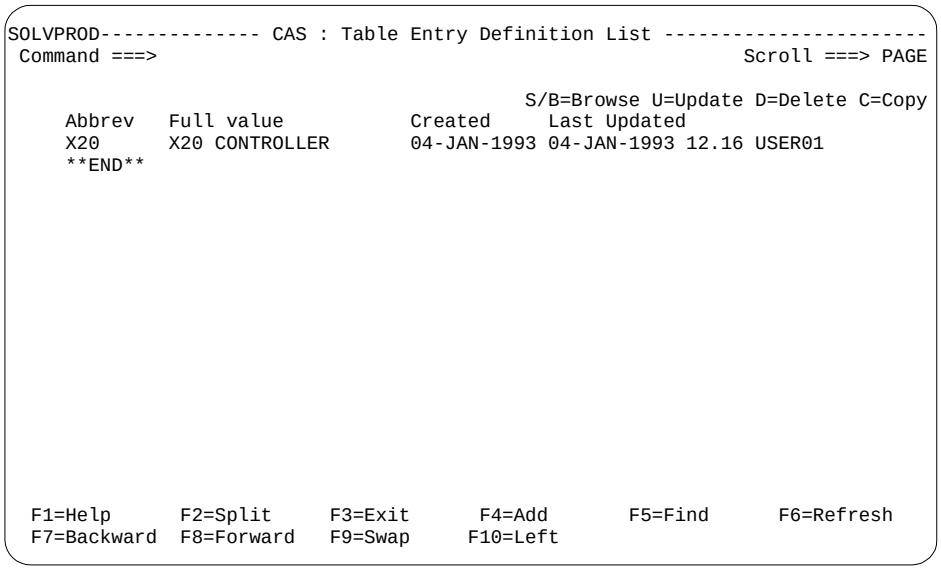


Figure 12-16. CAS : Table Entry Definition List (page 3)



The fields displayed on the Table Entry Definition List panel are as follows:

Abbrev

The abbreviated name of the table entry.

Note

This field is only displayed if you specify that abbreviated names are used for table entries. (See *Adding a Table Entry*, on page 12-17 for details).

Full Value
The full value of the table entry.

Description
A description of the table entry.

Note
This field is only displayed if you set the **Max description length** field in the table definition.

File ID
The identifier of the highest level MODS control file in which the table definition (and its entries) are present.

Seq
A number which determines the sequence in which table entries are displayed in a list.

Active
Whether the entry is available for use.

Created
The date the table entry was created.

Last Updated
The date and time that the table entry was last updated, and the user ID of the person who performed the update. If the entry has not been modified since installation, INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item. To add a new table entry, press the ADD key.

Mnemonic	Action
B, S, or /	Browse the selected table entry
U	Update the selected table entry
D	Delete the selected table entry
C	Copy the selected table entry

These options work in the same way as the corresponding menu options.

13

Maintaining Criteria

This chapter describes the Common Application Services (CAS) Criteria facility.

Criteria are rules that can be used in a test, for example, the expression on an &BOOLEXPR statement or the scan expression on an &NDBSCAN statement. They are used as a condition for the occurrence of some event—such as the inclusion of a record in a list, for example.

This chapter describes the how to define and maintain criteria definitions.

About Criteria

A criteria definition for a set of criteria consists of an identifier, a description of its behavior, a run time panel, a help name, a criteria exit, a data source, exit parameters, and the criterion condition.

A set of criteria can simply compare static values (for example, numbers or field names) or can be complex, combining numerous operators, attributes, connectors, parentheses, and values.

Criteria values can be variables (for example, the current date). When a set of criteria is recalled to perform a test, variable values are supplied either interactively by the user, using a run time panel, or by an exit procedure.

Criteria are used for the following purposes:

- Selection of objects for inclusion in a list
- Inclusion of objects in a report when using Report Writer
- The condition for display of a panel within a panel domain

Criteria definitions can be public or private. Public definitions are available to all users, while private definitions are available only to their owner.

Criteria can be predefined and reused. A criteria definition specifies the set of criteria and is named with a unique identifier. Criteria are stored on a database so that they can be recalled to provide the test that they define.

Run Time Panel

A criteria definition can include the identifier of a *run time panel*, to enable interactive entry of variable data for the criteria. This panel must request entry of data for variables within the set of criteria. See Chapter 6, *Maintaining Panels*, for details of how to define a panel.

Criteria Exit

You can specify a *criteria exit*, an NCL procedure to be executed during the process of building the set of criteria. The purpose of the criteria exit is to enable the initialization and editing of the run time panel and specification of variable data that is to be included in the set of criteria.

See Chapter 22, *Criteria Exit Procedure Interface*, for further details.

Data Source

You can specify a data source for the criteria definition. The purpose of this is to allow the criteria exit to determine the type of data to which the criteria are applied. This enables the definition of a general purpose criteria exit that can be used by multiple criteria definitions (since the exit determines the type of data and processes accordingly).

Exit Parameters

Exit parameters are passed to the criteria exit when it is executed in the process of building a set of criteria. A criteria exit can base its behavior on the exit parameters defined within a criteria definition. Thus it is possible to modify the behavior of a set of criteria by changing the exit parameters without having to change the exit procedure.

Substitution of Variable Data

The NCL built-in function &ZSUBST is used to substitute variable data into the criteria. The ampersand character (&) is used as the substitution character. The variables defined within the criteria can be set by the user through entry of data on a run time panel or by the criteria exit procedure.

Defining Criteria

You can create a criteria definition by adding a new definition (by using the appropriate menu option, or by using the ADD action from a list of criteria) or by copying an existing criteria definition.

Criteria Definition Menu

Criteria definitions are accessed through the CAS : Criteria Definition Menu (option **C** of the CAS : Maintenance Menu). This menu allows you to add, browse, update, delete, or copy criteria definitions, or to obtain a list of criteria definitions.

The Criteria Definition Menu is illustrated in Figure 13-1.

Figure 13-1. CAS : Criteria Definition Menu

```
SOLVPROD--------------- CAS : Criteria Definition Menu -----------------$CR010
Select Option ===>

   A   - Add Criteria
   B   - Browse Criteria
   U   - Update Criteria
   D   - Delete Criteria
   C   - Copy Criteria
   L   - List Criteria
   X   - Exit

Appl ID ........+ ___      ( Required B U D C  Optional A L )
Criteria Type ... _______  ( Required B U D C  Optional A L )
Userid .......... ________ ( Optional A B U D C L )
Criteria Name ... ________ ( Required B U D C  Optional A L )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the Criteria Definition Menu are as follows:

Appl ID
The 3-character identifier of the application to which the criteria belongs.

Criteria Type
A criteria type can be either:
- PUBLIC (the criteria is available to all users)
- PRIVATE (available only to the owner)

Userid
If the criteria type is PRIVATE, the user ID of the owner must be entered.

Criteria Name
The 1- to 8-character identifier of the criteria (alphanumeric and/or national characters). The criteria name uniquely identifies the criteria within the application and type.

When you select the list option, and you make full or partial entries in any of these fields, criteria definitions are only included in the list that matches the entries you make.

The following options are available on the Criteria Definition Menu:

Option	Explanation
A	Add a new criteria definition
B	Browse an existing criteria definition
U	Update an existing criteria definition
D	Delete a criteria definition
C	Copy an existing criteria definition
L	List criteria definitions

Detailed descriptions of these options follow.

Adding a Criteria Definition

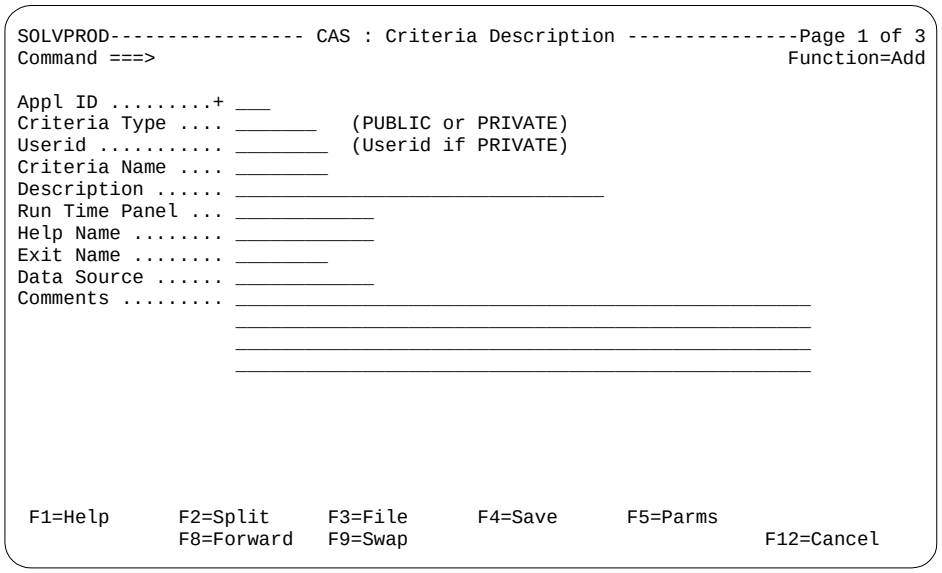
Defining a criteria involves completing a description panel, which identifies the criteria, and then specifying the actual criteria expression on a second panel.

You can also, optionally, define a series of exit parameters on a third panel if your criteria makes use of the criteria exit.

To define a criteria from the CAS : Criteria Definition Menu select option **A**.

This displays the CAS : Criteria Description panel, as shown in Figure 13-2.

Figure 13-2. CAS : Criteria Description Panel



The fields displayed on the CAS : Criteria Description panel are as follows:

Appl ID

The 3-character identifier of the application to which the criteria belongs.

Criteria Type

A criteria type can be either:

- PUBLIC (the criteria is available to all users)
- PRIVATE (available only to the owner)

Userid

If the criteria type is PRIVATE, the user ID of the owner must be entered.

Criteria Name

The 1- to 8-character identifier of the criteria (alphanumeric and/or national characters). The criteria name uniquely identifies the criteria within the application and type.

Description

A brief description of the behavior of the criteria (1 to 32 characters). The description is displayed when criteria are listed for selection purposes, and should thus be as informative as possible.

Run Time Panel

Enter the name of a panel that is used to solicit variable data from the user for criteria. The values entered by the user are used when evaluating the criteria.

The panel can be created using the MODS : Panel Maintenance facilities, as described in Chapter 6, *Maintaining Panels.*

Help Name

This (optional) field allows you to specify a function level help file to be presented when the HELP command is entered from the run time panel. Enter the function name of the appropriate function level help definition (1 to 12 characters).

If this field is left blank, the application level help for the application to which the criteria belongs is displayed in response to a help request.

See Chapter 10, *Maintaining Help*, for details of how to define this help definition.

Exit Name

This (optional) field allows you to specify an NCL procedure to be executed when the criteria is built. The procedure provides any variable data required by the criteria. Enter the name of an NCL procedure (1 to 8 characters). See Chapter 22, *Criteria Exit Procedure Interface*, for further details about the exit procedure.

Data Source

This (optional) field can be used by the exit procedure to determine the source of the data to which the criteria is to be applied.

This field identifies the data source to be passed to the criteria exit when it is executed during the process of building the criteria. The syntax of this field is dependent on the criteria exit.

Comments

A detailed description of the behavior of the criteria. Although this is an optional field, it is recommended that comments be included to assist the administrator when maintaining criteria definitions.

After completing the criteria description, press the FORWARD key to specify the actual criteria on the Criteria Text panel (as illustrated in Figure 13-3). A text editor is available for this purpose; see Appendix B, *Text Editor Commands*, for details of the commands that are available.

Figure 13-3. CAS : Criteria Text Panel

```
SOLVPROD--------------------- CAS : Criteria Text ------------------Page 2 of 3
 Command ===>                                    Function=Add Scroll ===> PAGE

 Appl ID ... ZPR      Type.Userid ... PUBLIC                 Name ... ASSIGN1

 LINE <---+----10---+----20---+----30---+----40---+----50---+----60---+----70-->
 **** ***************************** TOP OF DATA ******************************
 0001 STATUS NE ALL 'CLOSED','CANCELLED' AND ASSIGNEE EQ '&USERID'
 **** ***************************** BOTTOM OF DATA ***************************









  F1=Help      F2=Split     F3=File      F4=Save      F5=Find      F6=Change
  F7=Backward  F8=Forward   F9=Swap                              F12=Cancel
```

The criteria text panel is used to define the condition (or rule) for criteria. For example, the expression on an &BOOLEXPR statement or the scan expression on an &NDBSCAN statement.

The simplicity or complexity of the criteria and its syntax is dependent on what it is to be used for. For example, if it is to be used as the scan expression on an &NDBSCAN statement then it must obey the syntax rules for an &NDBSCAN scan expression.

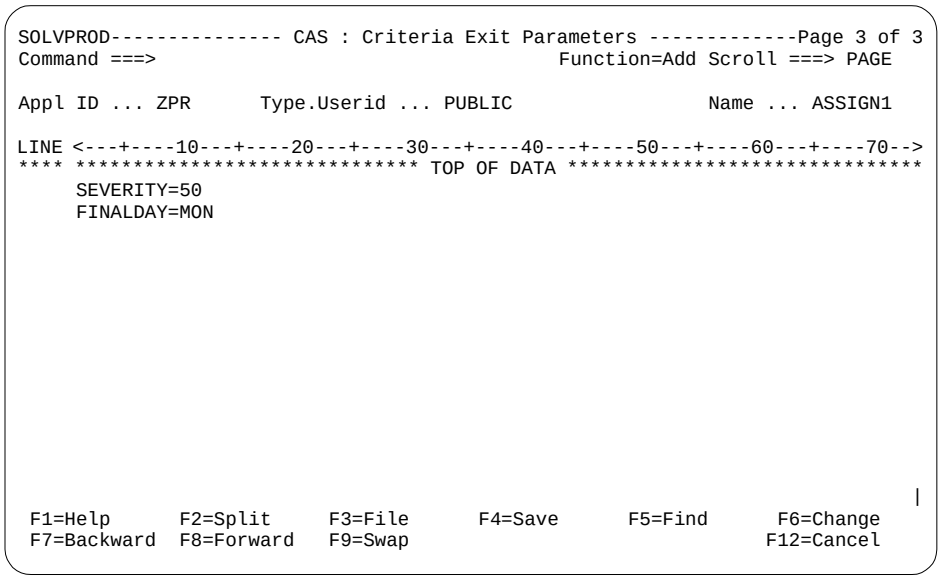
Criteria can consist of constant and variable data. Constant data never changes, for example, the identifier of an attribute. Variable data can change, for example, the current date. Variable data in the criteria is represented by an NCL variable, for example, &USERID. The NCL variables defined in the criteria, except system variables, must be set by the criteria exit or by the user entering data on the run time panel. The variable data in the criteria is substituted by the system when the criteria is built.

In the panel shown in Figure 13-3, the criteria specifies all problems which are not closed or cancelled that belong to the current user.

If the criteria uses an exit procedure, you can define exit parameters to be used by that procedure. To achieve this, press the PARMS key in the CAS : Criteria Description panel (shown in Figure 13-2).

After the above action, the CAS : Criteria Exit Parameters panel is displayed as shown in Figure 13-4.

Figure 13-4. CAS : Criteria Exit Parameters panel



This panel allows you to enter parameters for the exit procedure in the format required by the exit. A text editor is available for this purpose; see Appendix B, *Text Editor Commands*, for details of the commands that are available.

After specifying the criteria, press the FILE key. To cancel the definition, press the CANCEL key.

Maintaining Criteria Definitions

This section describes how to maintain existing criteria definitions.

You can specify the criteria you want to maintain by using the Criteria Definition Menu, and by completing the fields on the menu panel and selecting the appropriate option. Or you can specify criteria you want to maintain by selecting an equivalent action against a member of a list of criteria definitions.

Listing Criteria Definitions

Option **L** displays the criteria definitions over three panels, as shown in Figure 13-5, Figure 13-6 and Figure 13-7. Use the RIGHT and LEFT keys, to move between the panels.

Figure 13-5. CAS : Criteria Definition List (page 1)

```
SOLVPROD--------------- CAS : Criteria Definition List -----------------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Description                          File ID
   ZAM  PUB            ZAMAASCH  Asset-Agreement Relater Search       MODSDIS
   ZAM  PUB            ZAMAGSCH  Agreement General Search             MODSDIS
   ZAM  PUB            ZAMAG001  Agreement by Asset                   MODSUSR
   ZAM  PUB            ZAMAG002  Agreements Related to This Asset     MODSDIS
   ZAM  PUB            ZAMAPSCH  Asset-Person Relater Search          MODSDIS
   ZAM  PUB            ZAMASSCH  Asset General Search                 MODSDIS
   ZAM  PUB            ZAMAS001  Asset by Related Person              MODSDIS
   ZAM  PUB            ZAMAS002  Asset with Quantity = 1              MODSDIS
   ZAM  PUB            ZAMAS003  Asset by Agreement                   MODSDIS
   ZAM  PUB            ZAMAS004  Asset De-installation Life Cycle     MODSDIS
   ZAM  PUB            ZAMAS005  Asset Installation Life Cycle        MODSDIS
   ZAM  PUB            ZAMAS006  Asset License Search                 MODSDIS
   ZAM  PUB            ZAMAS007  Asset Line Search                    MODSDIS
   ZAM  PUB            ZAMAS008  Asset by Group                       MODSDIS
   ZAM  PUB            ZAMAS009  Asset by Expense                     MODSDIS
   ZAM  PUB            ZAMAS010  Asset Number                        MODSDIS
   ZAM  PUB            ZAMAS011  Asset by Product and Reference       MODSDIS
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 13-6. CAS : Criteria Definition List (page 2)

```
SOLVPROD--------------- CAS : Criteria Definition List -----------------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Run Time Panel  Help Name     Exit Name
   ZAM  PUB            ZAMAASCH  ZAMAASCH        CRITZAMAASCH  $OSPM21P
   ZAM  PUB            ZAMAGSCH  ZAMAGSCH        CRITZAMAGSCH  $OSPM21P
   ZAM  PUB            ZAMAG001  ZAMAG001        CRITZAMAG001  $OSPM21P
   ZAM  PUB            ZAMAG002                  CRITZAMAS002
   ZAM  PUB            ZAMAPSCH  ZAMAPSCH        CRITZAMAPSCH  $OSPM21P
   ZAM  PUB            ZAMASSCH  ZAMASSCH        CRITZAMASSCH  $OSPM21P
   ZAM  PUB            ZAMAS001  ZAMAS001        CRITZAMAS001  $OSPM21P
   ZAM  PUB            ZAMAS002                  CRITZAMAS002
   ZAM  PUB            ZAMAS003  ZAMAS003        CRITZAMAS003  $OSPM21P
   ZAM  PUB            ZAMAS004  ZAMAS004        CRITZAMAS004  $OSPM21P
   ZAM  PUB            ZAMAS005  ZAMAS005        CRITZAMAS005  $OSPM21P
   ZAM  PUB            ZAMAS006  ZAMAS006        CRITZAMAS006  $OSPM21P
   ZAM  PUB            ZAMAS007  ZAMAS007        CRITZAMAS007  $OSPM21P
   ZAM  PUB            ZAMAS008  ZAMAS008        CRITZAMAS008  $OSPM21P
   ZAM  PUB            ZAMAS009  ZAMAS009        CRITZAMAS009  $OSPM21P
   ZAM  PUB            ZAMAS010                  CRITZAMAS010
   ZAM  PUB            ZAMAS011                  CRITZAMAS011
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 13-7. CAS : Criteria Definition List (page 3)

```
SOLVPROD--------------- CAS : Criteria Definition List -----------------------
Command ===>                                                 Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Appl Typ  Userid    Name      Created     Last Updated
   ZAM  PUB            ZAMAASCH  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAGSCH  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAG001  20-FEB-1993 12-MAY-1993 10.30 USER01
   ZAM  PUB            ZAMAG002  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAPSCH  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMASSCH  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS001  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS002  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS003  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS004  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS005  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS006  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS007  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS008  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS009  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS010  20-FEB-1993 20-FEB-1993 00.00 INSTALL
   ZAM  PUB            ZAMAS011  20-FEB-1993 20-FEB-1993 00.00 INSTALL
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left
```

The fields displayed on the CAS : Criteria Definition List panel are as follows:

Appl
The application identifier of the criteria.

Typ
The type of the list—PRI (private) or PUB (public).

Userid
The user ID of the owner of the list if it is a private list.

Name
The identifier of the criteria definition.

Description
A description of the purpose of the criteria.

File ID
The identifier of the MODS control file in which the definition is held.

Run Time Panel
If it is used, the identifier of the run time panel for the criteria.

Help Name
The identifier of the help definition (if defined) for the criteria.

Exit Name
The identifier of the criteria exit procedure.

Created
The date that the criteria definition was created.

Last Updated
The date and time that the criteria definition was last modified and the user ID of the person who performed the update. If the criteria has not been modified since installation, INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
S, B or /	Browse the selected criteria definition
U	Update the selected criteria definition
D	Delete the selected criteria definition
C	Copy the selected criteria definition

These actions are described below. You can also perform these functions by choosing the corresponding options from the CAS : Criteria Definition Menu.

Browsing a Criteria Definition

Select option **B** to browse a criteria definition. The three panels of the criteria definition are displayed as shown in Figure 13-2, Figure 13-3, and Figure 13-4, except that the function is Browse and the fields are displayed as output fields and cannot be modified.

Use the FORWARD, BACKWARD, and PARMS commands to move between the panels.

Updating a Criteria Definition

Select option **U** to update a criteria definition. The three panels of the criteria definition are displayed as shown in Figure 13-2, Figure 13-3, and Figure 13-4, except that the function is Update and the **Appl ID**, **Criteria Type**, **User ID** and **Criteria Name** fields cannot be modified.

Update the other fields as required; refer to the section titled *Adding a Criteria Definition*, on page 13-4 for details.

After updating the criteria definition, press the FILE key. To cancel the update, press the CANCEL key.

Deleting a Criteria Definition

Select option **D** to delete a criteria definition. A message is displayed requiring you to confirm that you want to proceed.

Press ENTER to delete the criteria definition, or press the CANCEL key to cancel the deletion.

Copying a Criteria Definition

To copy an existing criteria definition to another (new) criteria definition, select option **C** from the Criteria Definition Menu and provide identification details of the criteria definition to copy from (or use the appropriate action in a list).

The three panels of the new criteria definition are displayed, as shown in Figure 13-2, Figure 13-3 and Figure 13-4, with the fields copied from the nominated criteria definition. Modify these as required—see *Adding a Criteria Definition*, on page 13-4 for field details.

To complete the copy, press the FILE key. To cancel the copy, press the CANCEL key.

14

Maintaining Commands

This chapter describes the Common Application Services (CAS) Commands facility. This facility provides you with the ability to define commands to perform your own specialized functions.

About Commands

A command definition contains identifying information about the command and specifies an action that occurs when the command is executed.

Unlike the other CAS components, commands are not defined as belonging to a particular application. This means that all commands defined to CAS can be used by all applications.

Defining a Command

You can define a command in two ways; either by adding a new command definition by selecting the Add Command option from the Command Definition Menu (or pressing the ADD key on a list of commands), or by copying an existing definition.

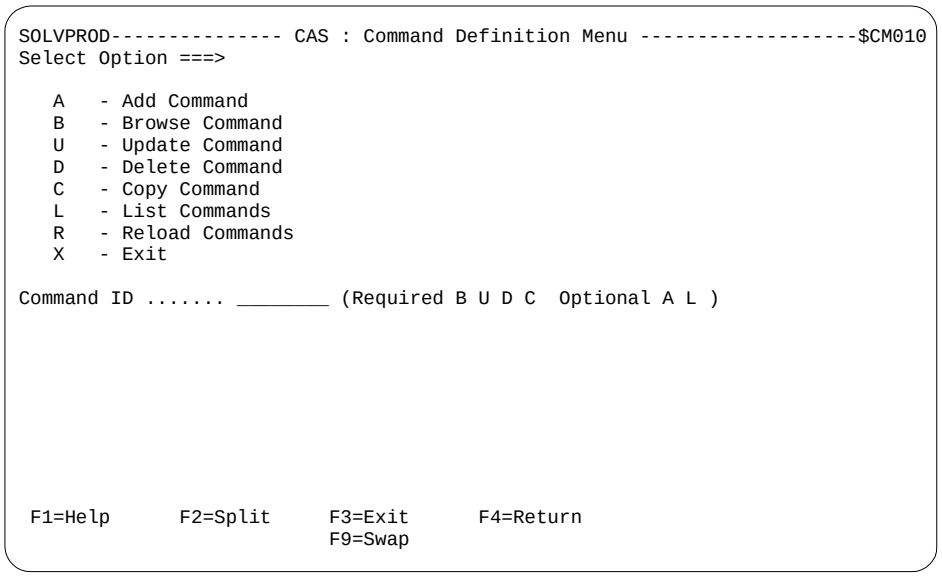
For details of how to copy an existing command definition see *Copying a Command Definition*, on page 14-9.

Command Definition Menu

Commands are accessed through the CAS : Command Definition Menu (option **CM** of the CAS : Maintenance Menu).

The Command Definition Menu is illustrated in Figure 14-1.

Figure 14-1. CAS : Command Definition Menu



The field displayed on the Command Definition Menu is as follows:

Command ID

This (up to) 8 character name uniquely identifies the command.

The ***Command ID*** is entered on the command line by the user to invoke the command. You should use a meaningful word or mnemonic to identify the command.

The following options are available on the Command Definition Menu:

Option	Explanation
A	Add a new command definition
B	Browse an existing command definition
U	Update an existing command definition
D	Delete an existing command definition
C	Copy an existing command definition
L	List command definitions
R	Reload command definitions

Note

Whenever you add a new command or modify existing commands you must execute the Reload Commands option for your changes to take effect. See *Reloading Command Definitions*, on page 14-9.

A detailed description of these options follows.

Adding a Command Definition

Option **A** displays the Command Definition panel, shown in Figure 14-2.

Figure 14-2. CAS : Command Definition Panel

```
SOLVPROD------------------ CAS : Command Definition ---------------- Page 1 of 1
 Command ===>                                                  Function=Add

 Command ID ... mycmd

 Description .. Execute the named procedure_______

 Comments ..... ________________________________________________________
                ________________________________________________________
                ________________________________________________________
                ________________________________________________________

 Action ....... EXEC &$CMPARM1_____________________________________________________
                ____________________________________________________________________
                ____________________________________________________________________
                ____________________________________________________________________









  F1=Help      F2=Split     F3=File      F4=Save
                            F9=Swap                              F12=Cancel
```

The fields displayed on the CAS : Command Definition panel are as follows:

Command ID

The identifier of the command.

The ***Command ID*** is entered on the command line by the user to invoke the command. You should use a meaningful word or mnemonic to identify the command.

Description

A 1 to 32 character description of the command.

Comments

A more detailed description of the behavior of the command. Although this is an optional field, it is recommended that comments be included to assist the CAS administrator when maintaining command definitions.

Action

Specifies the action to be taken when this command is entered.

An Action consists of any item in Table 14-1, or a word recognized by the calling procedure, together with any required parameters. When the command is entered and passed to CAS (through $CACALL) by the controlling NCL procedure, CAS either:

- Performs the required action (if the action is from Table 14-1)
- Returns the unknown command (and any parameters) to the calling procedure

The parameters can include constant or variable data, or both. Variable data consists of NCL variables that are set by CAS when the command is processed.

Valid variables are:

- **&$CMPARMS**
  Contains all data entered by the user following the command.
- **&$CMPARM1 ... &$CMPARM64**
  Contains the command parameters (delimited by a space) entered by the user following the command.

Table 14-1. Actions Supported by CAS

Command	Action
CMD [*command*]	Enter the Command Entry facility and prime the command field with the *command*.
EXEC *procname* [*parameters*]	Execute the procedure *procname*, with specified parameter(s)
DISCONN	Disconnect the session
HELP	Display online Help information
KEYS [PRI | ALT | OFF | SET]	PRI Build and return Primary function key area. ALT Build and return Alternate function key area. OFF Turn function key display off. SET Allow user to set function keys 13-24. The default is to toggle between PRI and ALT.
LOCK	Lock the session
NOTEPAD	Invokes the CAS notepad facility
PASSWORD	Allow user to change password and/or user details
PQUEUE [user ID]	List queued print requests for *user ID.* Defaults to your user ID and can be a generic ID if terminated by an asterisk.

Table 14-1. Actions Supported by CAS

Command	Action
PSKIP [*options*]	Jump to the specified panel (equivalent to the '=' panel skip command)
RETRIEVE [? prefix]	Retrieve the last command. ? is for prompt support. If followed by a prefix, obtains a list of commands starting with that prefix.
SPLIT	Split the window at the current cursor position
START *procname [parameters]*	Start NCL procedure *procname* with specified *parameters*
SWAP	Swap to the other window
WHERE	Display current NCL procedure details

After completing the command definition, press the FILE key. To cancel the command definition, press the CANCEL key.

Maintaining Command Definitions

This section discusses the facilities for maintaining existing command definitions. You can modify existing command definitions by selecting the appropriate menu option or by executing an action against an entry in a command list.

Listing Command Definitions

Select option **L** to produce a list of existing command definitions shown over three panels, as shown in Figure 14-3, Figure 14-4, and Figure 14-5. Use the RIGHT and LEFT keys to scroll between the panels.

If you make an entry (or partial entry) in the **Command ID** field when you select this option, the list is constrained to command definitions that match the entry you made.

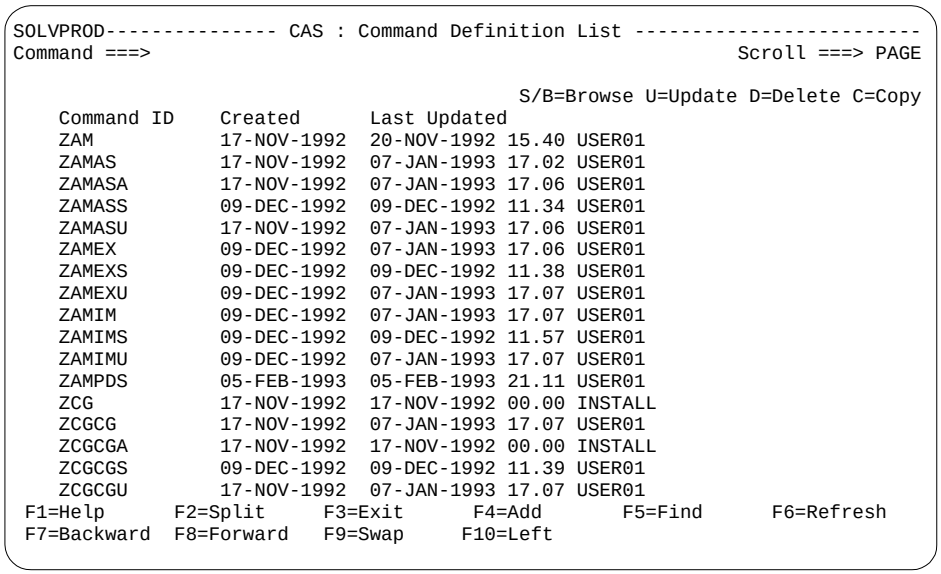
Figure 14-3. CAS : Command Definition List (page 1)

```
SOLVPROD--------------- CAS : Command Definition List -------------------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
    Command ID    Description                                          File ID
    ZAM           Invoke SOLVE:Asset                                   MODSTST
    ZAMAS         Browse Asset                                         MODSUSR
    ZAMASA        Add Asset                                            MODSUSR
    ZAMASS        Search Assets                                        MODSTST
    ZAMASU        Update Asset                                         MODSUSR
    ZAMEX         Browse Expense                                       MODSUSR
    ZAMEXS        Search Expenses                                      MODSTST
    ZAMEXU        Update Expense                                       MODSUSR
    ZAMIM         Browse Impact                                        MODSUSR
    ZAMIMS        Search Impacts                                       MODSTST
    ZAMIMU        Update Impact                                        MODSUSR
    ZAMPDS        Search Products                                      MODSUSR
    ZCG           Invoke SOLVE:Change                                  MODSDIS
    ZCGCG         Browse Change                                        MODSUSR
    ZCGCGA        Add Change                                           MODSDIS
    ZCGCGS        Search Changes                                       MODSTST
    ZCGCGU        Update Change                                        MODSUSR
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 14-4. CAS : Command Definition List (page 2)

```
SOLVPROD--------------- CAS : Command Definition List -------------------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
    Command ID    Action
    ZAM           EXEC ZMGCALL OPT=PMENU APPL=ZAM PSKIP='&$CMPARMS'
    ZAMAS         EXEC $OSCALL SERVICE=ACTION ACTION=ZOSBROWSE CLASS=ZAMASSET N
    ZAMASA        EXEC $OSCALL SERVICE=ACTION ACTION=ZOSADD CLASS=ZAMASSET
    ZAMASS        EXEC $CACALL OPT=ACTION ACTION=DISPLAY CLASS=LIST NAME='APPL=
    ZAMASU        EXEC $OSCALL SERVICE=ACTION ACTION=ZOSUPDATE CLASS=ZAMASSET N
    ZAMEX         EXEC $OSCALL SERVICE=ACTION ACTION=ZOSBROWSE CLASS=ZAMEXP NAM
    ZAMEXS        EXEC $CACALL OPT=ACTION ACTION=DISPLAY CLASS=LIST NAME='APPL=
    ZAMEXU        EXEC $OSCALL SERVICE=ACTION ACTION=ZOSUPDATE CLASS=ZAMEXP NAM
    ZAMIM         EXEC $OSCALL SERVICE=ACTION ACTION=ZOSBROWSE CLASS=ZAMIMP NAM
    ZAMIMS        EXEC $CACALL OPT=ACTION ACTION=DISPLAY CLASS=LIST NAME='APPL=
    ZAMIMU        EXEC $OSCALL SERVICE=ACTION ACTION=ZOSUPDATE CLASS=ZAMIMP NAM
    ZAMPDS        EXEC $CACALL OPT=ACTION ACTION=DISPLAY CLASS=LIST NAME='APPL=
    ZCG           EXEC ZMGCALL OPT=PMENU APPL=ZCG PSKIP='&$CMPARMS'
    ZCGCG         EXEC $OSCALL SERVICE=ACTION ACTION=ZOSBROWSE CLASS=ZCGCHNG NA
    ZCGCGA        EXEC $CACALL OPT=ACTION ACTION=DISPLAY CLASS=LIST NAME='APPL=
    ZCGCGS        EXEC $CACALL OPT=ACTION ACTION=DISPLAY CLASS=LIST NAME='APPL=
    ZCGCGU        EXEC $OSCALL SERVICE=ACTION ACTION=ZOSUPDATE CLASS=ZCGCHNG NA
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 14-5. CAS : Command Definition List (page 3)



```
SOLVPROD--------------- CAS : Command Definition List ------------------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Command ID    Created      Last Updated
   ZAM           17-NOV-1992  20-NOV-1992 15.40 USER01
   ZAMAS         17-NOV-1992  07-JAN-1993 17.02 USER01
   ZAMASA        17-NOV-1992  07-JAN-1993 17.06 USER01
   ZAMASS        09-DEC-1992  09-DEC-1992 11.34 USER01
   ZAMASU        17-NOV-1992  07-JAN-1993 17.06 USER01
   ZAMEX         09-DEC-1992  07-JAN-1993 17.06 USER01
   ZAMEXS        09-DEC-1992  09-DEC-1992 11.38 USER01
   ZAMEXU        09-DEC-1992  07-JAN-1993 17.07 USER01
   ZAMIM         09-DEC-1992  07-JAN-1993 17.07 USER01
   ZAMIMS        09-DEC-1992  09-DEC-1992 11.57 USER01
   ZAMIMU        09-DEC-1992  07-JAN-1993 17.07 USER01
   ZAMPDS        05-FEB-1993  05-FEB-1993 21.11 USER01
   ZCG           17-NOV-1992  17-NOV-1992 00.00 INSTALL
   ZCGCG         17-NOV-1992  07-JAN-1993 17.07 USER01
   ZCGCGA        17-NOV-1992  17-NOV-1992 00.00 INSTALL
   ZCGCGS        09-DEC-1992  09-DEC-1992 11.39 USER01
   ZCGCGU        17-NOV-1992  07-JAN-1993 17.07 USER01
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left
```

The fields displayed on the Command Definition List are as follows:

Command ID
This (up to) 8-character name uniquely identifies the command.

Description
A description of the command.

File ID
The identifier of the MODS control file in which the command definition is stored.

Action
The action associated with the command.

Created
The date that the command definition was created.

Last Updated
The time and date that the command definition was last updated and the user ID of the person who performed the update. If the definition has not been modified since it was installed, INSTALL is displayed.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item.

Mnemonic	Action
S, B, or /	Browse the selected command definition
U	Update the selected command definition
D	Delete the selected command definition
C	Copy the selected command definition

You can also add a new command definition by pressing the ADD key on a list of command definitions.

These functions can also be performed by choosing the corresponding option from the Command Definition Menu.

Browsing a Command Definition

Option **S**, **B**, or **/** displays the Command Definition panel, as shown in Figure 14-2, except that the function is Browse and fields cannot be modified.

Updating a Command Definition

Option **U** displays the Command Definition panel, as shown in Figure 14-2, except that the function is Update and the **Command ID** field cannot be modified. Update this as required; refer to the section titled *Adding a Command Definition*, on page 14-3 for details.

After updating the command definition, press the FILE key. To cancel the update, press the CANCEL key.

Deleting a Command Definition

Option **D** displays the Command Definition panel, showing the specified command, with a message requiring you to confirm that you want to proceed with the deletion.

Press ENTER to delete the command definition, or the CANCEL key to cancel the deletion.

Copying a Command Definition

To copy an existing command definition to another (new) command definition, select option **C**.

This displays the Command Definition panel, as shown in Figure 14-2 showing the copied command definition. Update the new command definition as required (see the section titled *Adding a Command Definition*, on page 14-3 for details).

After copying the command definition, press the FILE key. To cancel the copy, press the CANCEL key.

Reloading Command Definitions

Select option **R** from the Command Definition Menu to reload command definitions when you make a change to one or more commands.

A reload should be performed when any command maintenance is carried out—if a reload is not performed, the changes do not take effect until the next time your SOLVE system is started.

15

Printing MODS Components

This chapter describes how to print MODS component reports by using option **REP** from the MODS : Primary Menu.

About MODS Component Reports

The report types that can be obtained using option **REP** are as follows:

- Application definitions
- CAS components, including:
  - Messages
  - Help
  - Menus
  - Lists
  - Criteria
  - Tables
  - Panel domains
  - Commands
- Object Class specifications
- Object Class definitions
- Print Services Manager (PSM) definitions
- Report Writer report definitions

Note

Help text and message text can also be printed by other means.

To find out how to print help text in either of two different formats by using the various CAS : Help Definition menus, refer to Chapter 10, *Maintaining Help*.

To find out how to print message text by using the CAS : Message Definition Menu, refer to Chapter 11, *Maintaining Messages*.

You *cannot* use option **REP** to print the following:

- Map definitions and their ASN.1 source code
- Panel definitions

Refer to Chapter 25, *Maintaining Maps*, on how to print map definitions and their ASN.1 source code. Refer to Chapter 6, *Maintaining Panels*, on how to print panel definitions.

Printing Components Using the REP Option

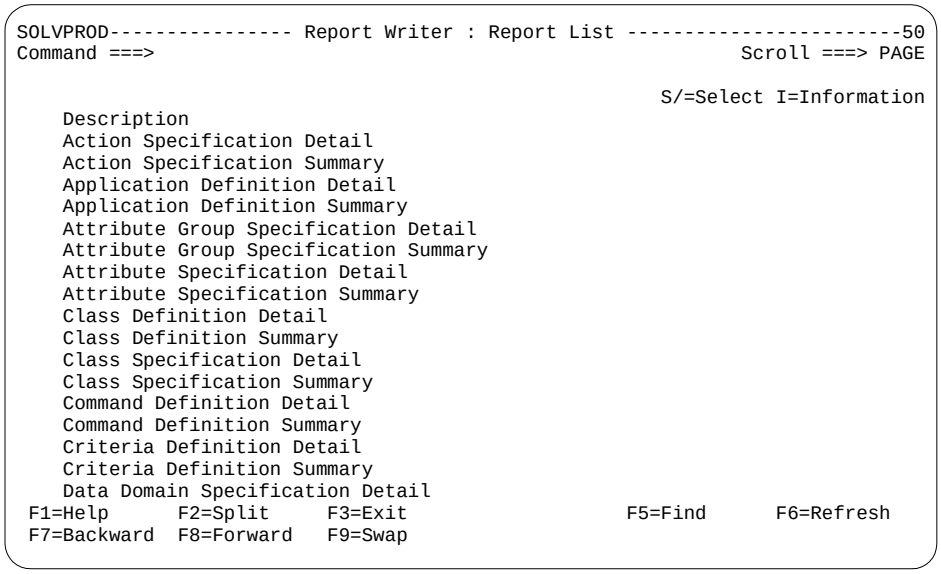
You can print MODS component reports by selecting option **REP** from the MODS : Primary Menu. The Report Writer : Report List panel appears (Figure 15-1). The panel lists the component reports that you can print. For each report, you can select the definitions to be included.

There are two report formats for each component: detail and summary. A detail report contains exact copies of the selected definitions within a component. A summary report contains single line descriptions of the selected definitions within a component.

Note

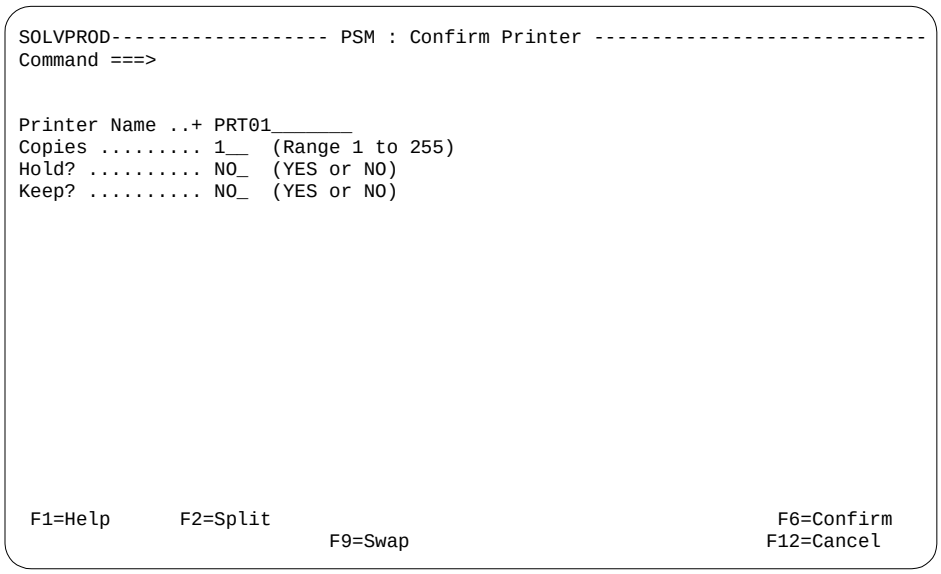
If you want to find out more about a report, type **I** next to the report and press ENTER to access the Report Writer : Report Information panel. The panel identifies the report and the base criteria that determine the contents of the report.

Figure 15-1. Report Writer : Report List



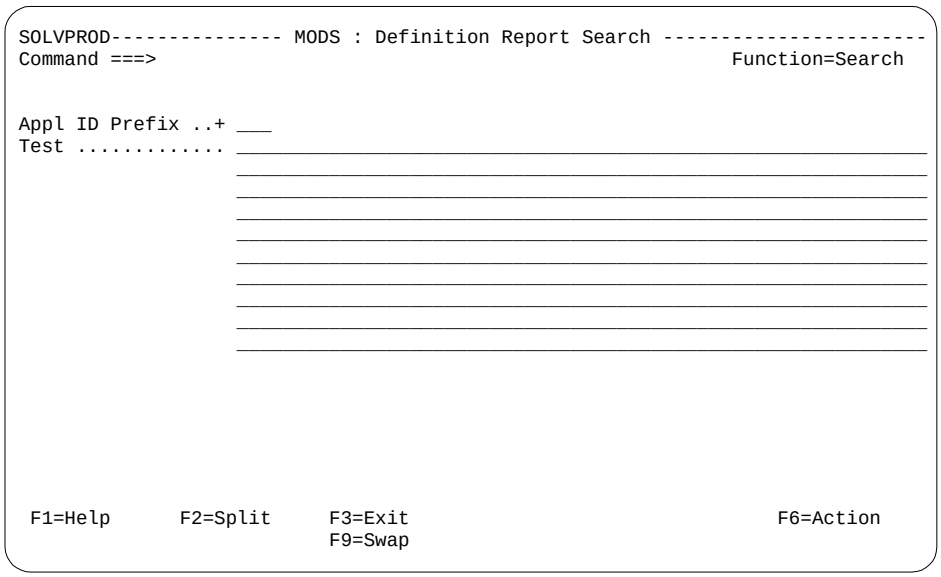
To select a report for printing, type **S** or / next to the report and press ENTER. The PSM : Confirm Printer panel appears (Figure 15-2).

Figure 15-2. PSM : Confirm Printer



If necessary, change the values in the fields, then press the CONFIRM key. The MODS : Definition Report Search panel appears (Figure 15-3). Use the panel to restrict the definitions you want to include in the report. If you do not fill in any fields, all definitions of the selected component are printed.

Figure 15-3. MODS : Definition Report Search



Note

Depending on the selected component, the panel in Figure 15-3 might have a different title and first field. For example, the title is MODS : Command Definition Report Search and the field is **Command ID Prefix** for command definitions.

Use the prefix field to restrict the definitions to be included in the report for a particular MODS component: for example, $LH indicating the CAS list definitions. Use the **Test** fields to further restrict the definitions by a Boolean expression. Refer to *Network Control Language Reference* for the syntax of Boolean expressions. The expression tests for certain values within a definition. The following example restricts the report to private list definitions belonging to USER01:

```
&$LHLDTYPE='PRIVATE' AND &$LHLDUSERID='USER01'
```

See the next section, *Identifying Definition Variables*, on how to identify the variables that hold those values.

Note

If reports for help or messages are selected, the panel in Figure 15-3 will contain a language code field. The language code is that of the records for which you want to produce the report. If this field is left blank, the report will be produced for English language help or messages.

After you specify the restrictions, press the ACTION key to print the report.

If the report is on hold, you can use the **PQ**[UEUE] command to access the PSM output queue and view the report.

Identifying Definition Variables

To use the **Test** fields on the MODS definition report search panels, you need to know the variables that hold the definition field values. Access the required list of variables as follows:

Step 1. Use the **=D.REP** path specification to access the list of MODS component reports. The Report Writer : Report List panel appears (Figure 15-4).

Figure 15-4. Report Writer : Report List

```
SOLVPROD---------------- Report Writer : Report List ------------------------50
Command ===>                                                  Scroll ===> PAGE

                                                        S/=Select I=Information
    Description
    Action Specification Detail
    Action Specification Summary
    Application Definition Detail
    Application Definition Summary
    Attribute Group Specification Detail
    Attribute Group Specification Summary
    Attribute Specification Detail
    Attribute Specification Summary
    Class Definition Detail
    Class Definition Summary
    Class Specification Detail
    Class Specification Summary
    Command Definition Detail
    Command Definition Summary
    Criteria Definition Detail
    Criteria Definition Summary
    Data Domain Specification Detail
 F1=Help      F2=Split     F3=Exit                  F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap
```

Step 2. Type **I** next to the report you require and press ENTER. The Report Writer : Report Information panel appears. The following example (Figure 15-5) provides the information on the detailed report for the CAS list definitions:

Figure 15-5. Report Writer : Report Information

```
SOLVPROD------------ Report Writer : Report Information -----------Page 1 of 1
Command ===>

Report Details
  Report Appl ... $ADLH
  Type .......... PUBLIC                  Userid ...
  Name .......... $DETAIL                 Group ....
  Description ... List Definition Detail
  Comments ......



Criteria Details
  Appl ID ....... $AD
  Type .......... PUBLIC                  Userid ...
  Name .......... $ADRWSCH
  Description ... MODS Report Search
  Comments ...... User specified search of MODS definitions.




 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

Step 3. Note the value in the **Report Appl** field: for example, $ADLH for the CAS list definitions.

Step 4. Type **=D.C.T.E** at the Command ===> prompt and press ENTER. The CAS : Table Entry Definition Menu panel appears (Figure 15-6).

Figure 15-6. CAS : Table Entry Definition Menu

```
SOLVPROD------------ CAS : Table Entry Definition Menu ----------------$VM021
Select Option ===>

   A   - Add Table Entry
   B   - Browse Table Entry
   U   - Update Table Entry
   D   - Delete Table Entry
   C   - Copy Table Entry
   L   - List Table Entries
   R   - Reload Tables
   X   - Exit

Appl ID .........+ ___          ( Required all )
Field Name ......+ _____________ ( Required A B U D C L )
Full Value ....... _____________ ( Required B U D C )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

Step 5. Type **L** at the Select Option ===> prompt and **$AD** in the **Appl ID** field. Append DATA to the value noted in Step 3 (for example, $ADLHDATA for the CAS list definitions), and type the new value in the **Field Name** field. Press ENTER. The CAS : Table Entry Definition List panel appears. The following example (Figure 15-7) shows a partial list of the variables for the CAS list definitions:

Figure 15-7. CAS : Table Entry Definition List

```
SOLVPROD------------- CAS : Table Entry Definition List -----------------------
Command ===>                                                 Scroll ===> PAGE

                                        S/B=Browse U=Update D=Delete C=Copy
    Full value           Description                                  File ID
    $DB@DBID             File ID                                      MODSUSR
    $LHLDACTC            Action List Comment Line                     MODSUSR
    $LHLDADD             Add Allowed?                                 MODSUSR
    $LHLDAPPL            Appl ID                                      MODSUSR
    $LHLDAUTORAT         Auto Refresh Rate                            MODSUSR
    $LHLDCOM*            Comments                                     MODSUSR
    $LHLDCRITAPL         Criteria Appl ID                             MODSUSR
    $LHLDCRITNAM         Criteria Name                                MODSUSR
    $LHLDCRITTYP         Criteria Type                                MODSUSR
    $LHLDCRITUSR         Criteria Userid                              MODSUSR
    $LHLDCRTDATE         Creation Date                                MODSUSR
    $LHLDCRTTIME         Creation Time                                MODSUSR
    $LHLDCRTUSER         Creation User ID                             MODSUSR
    $LHLDDATASRC         Data Source                                  MODSUSR
    $LHLDDEFALTM         Default Mnemonic                             MODSUSR
    $LHLDDESC            Description                                  MODSUSR
    $LHLDEMPTY           Present Empty List?                          MODSUSR
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                 F11=Right
```

The panel lists the names of the definition variables in the **Full value** column with the corresponding field names in the **Description** column. You use these variables in the **Test** fields of the MODS definition report search panel.

Part III

MODS Administration

16

Maintaining Panel Libraries

A panel library contains a series of panel definitions. This chapter describes the facilities for:

- Defining and maintaining panel libraries
- Copying panels between libraries on different paths

About Panel Libraries

Panels are stored in VSAM datasets for fast retrieval and update. A VSAM dataset containing panel definitions is called a *panel library.* SOLVE supports multiple panel libraries.

Individual panel definitions are referred to as *library members*. To maintain panel definitions, refer to Chapter 6, *Maintaining Panels.*

A library can be used as the sole source of panel definitions, or it can be concatenated with other libraries defined to the system. A concatenation of libraries is called a *panel path.* Each user can be defined to use a different path. The default path is called PANELS.

Panel paths are further described in the chapter titled *SOLVE Access and Security* in the *SOLVE Management Services Implementation and Administration Guide.*

Maintaining Panel Libraries

This section describes the facilities available for maintaining panel libraries.

Accessing the Panel Library Maintenance Facilities

The panel library maintenance facilities are accessed through the MODS : Panel Library Maintenance Menu, by selecting the following:

- Option **D** from the SOLVE : Primary Menu
- Option **AD** from the MODS : Primary Menu
- Option **P** from the MODS : Administration Menu

The MODS : Panel Library Maintenance Menu is illustrated in Figure 16-1.

Figure 16-1. MODS : Panel Library Maintenance Menu

```
SOLVPROD------------ MODS : Panel Library Maintenance Menu ---------------------
 Select Option ===>

    C   - Copy Panel(s)
    L   - Library Definitions
    X   - Exit


 'From' Library ..........+ ________     ( Required C )

 'To' Library ............+ ________     ( Required C )

 Panel Name ............... ____________ ( Blank, Full or Generic name,
                                           e.g. '*'  for all panels, or
                                                'D*' for all starting with D )

 Replace Like-Named Panels? NO_          ( Required C YES or NO )

 Copy All Matching Panels?  NO_          ( Required C YES or NO )



  F1=Help      F2=Split     F3=Exit      F4=Return
                            F9=Swap
```

The MODS : Panel Library Maintenance Menu allows you to:

- Copy panel definitions from one library to another (regardless of which paths the libraries are in), as described below.
- Access the MODS : Library Definition Menu to maintain library definitions, as described on page 16-6.

Copying Panels Between Libraries (Any Path)

Select option **C** from the MODS : Panel Library Maintenance Menu to copy a panel library. You need to enter the following data:

From Library
The (up to) 8-character name of the panel library from which to copy panels

To Library
The (up to) 8-character name of the panel library to which to copy panels

Panel Name
The name of the panel to be copied. You can specify the full name to copy a particular panel, or leave the field blank for a selection list of panels in the **From Library**.

The panel name can also be specified generically, using the asterisk character (*) as a *wildcard* match for the panel names. For example:

ABC
Matches the panel named ABC.

A*C
Matches any panel with a three letter name starting with A and ending with C.

ABC*
Matches any panel whose name starts with the letters ABC.

***
Matches every panel in the library.

Specifying a generic name works in conjunction with the specification of the **Copy All Matching Panels** field (see below).

The following two fields affect the way the Copy operates:

Replace Like-Named Panels
Determines how panels are copied to the **To Library**, when panels of the same name exist in both the **To** and the **From** libraries.

- Enter **YES** to replace panels in the **To Library** with panels of the same name in the **From Library**.
- Enter **NO** to prevent panels in the **To Library** from being replaced by panels of the same name in the **From Library**. **NO** is the default value for this field.

  Specifying **NO** safeguards the existing contents of the **To Library** against unintentional erasure.

Copy All Matching Panels

Determines what happens when a generic **Panel Name** (see above) is entered.

- Enter **YES** to copy all panels in the **From Library** which match the generic specification to the **To Library** (subject to the **Replace Like-Named Panels** option).
- Enter **NO** to display a selection list of panels in the **From Library** which match the generic name. **NO** is the default value for this field.

After completing the fields on the MODS : Panel Library Maintenance Menu, press ENTER. The Copy is performed as follows:

- If you specified one particular panel in the **Panel Name** field, or entered **YES** in the **Copy All Matching Panels** field, the Copy is performed when you press ENTER.
- If you left the **Panel Name** blank, or specified a generic panel name and entered **NO** in the **Copy All Matching Panels** field, the MODS : Panel Copy List is displayed when you press ENTER. Use this list to select the actual panels to be copied from the **From Library**.

Selecting Panels to Copy

The MODS : Panel Copy List is shown in Figure 16-2. This list displays panels in the **From Library** which generically match the **Panel Name** you specified (or all panels in the **From Library**, if you left **Panel Name** blank).

The top left of the display shows the name of the **From Library** (LODGED) and the **To Library** (TEST).

Figure 16-2. MODS : Panel Copy List

```
LODGED TO TEST------------ MODS : Panel Copy List ------------------------COPY
Command ===>                                                 Scroll ===> PAGE

                                                 S/C=Copy R=Replace B=Browse
    Name                  Created      Modified          Size  Mlev    Id
    $ADEX20MC             20-NOV-1992  20-NOV-1992 15.08   30     6    USER01
    $DDCOMPP              26-NOV-1992  26-NOV-1992 15.09   62     0    USER01
    $DDEQUEP              26-NOV-1992  26-NOV-1992 15.09   62     0    USER01
    $DDIMPEP              26-NOV-1992  26-NOV-1992 15.09   62     0    USER01
    $DDMAPEP              26-NOV-1992  26-NOV-1992 15.09   62     0    USER01
    $DDTAGEP              26-NOV-1992  26-NOV-1992 15.09   62     0    USER01
    $DDTYPEP              26-NOV-1992  26-NOV-1992 15.09   62     0    USER01
    $EASIUPD2             01-NOV-1992  25-NOV-1992 11.31   35     1    USER01
    $EDTEXTEDIT           01-NOV-1992  19-NOV-1992 12.04   88     3    USER01
    $OSDD5SIATTR          17-NOV-1992  10-DEC-1992 10.36   33     2    USER01
    $OSDD5SIRECS          17-NOV-1992  10-DEC-1992 10.36   35     2    USER01
    $OSDD5SISTAT          17-NOV-1992  10-DEC-1992 10.36   33     2    USER01
    $OSDD5TIATTR          16-NOV-1992  10-DEC-1992 10.37   33     3    USER01
    $OSDD5TIRECS          16-NOV-1992  10-DEC-1992 10.39   35     4    USER01
    $OSDD5TISTAT          16-NOV-1992  10-DEC-1992 10.37   33     4    USER01
    $OSND6HIATTR          07-DEC-1992  07-DEC-1992 14.37   27     3    USER01
    $OSOMSRSTAT           20-NOV-1992  23-NOV-1992 16.54   45     9    USER01
 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap
```

The fields displayed on the Panel Copy List are as follows:

Name
The identifier of the panel library.

Created
The date the panel library was created.

Modified
The time and date that the library was last modified.

Size
The size of the panel library.

Mlev
The modification level for the panel. Each time the panel is modified this number is incremented by 1 (to a maximum of 99).

Id
The identifier of the user that last updated the panel library.

You can perform the following actions on any panel in the list, by using the appropriate mnemonic:

Mnemonic	Action
S (or C)	Copy the selected panel.
R	Replace the selected panel. You can use this option to overwrite panels in the **To Library** that are protected by the **Replace Like-Named Panels** field being set to **NO**.
B	Browse the selected panel definition.

You can select panels for copying by placing an **S**, **C**, or **R** beside them and pressing ENTER.

After copying all required panels, press the EXIT key to return to the MODS : Panel Library Maintenance Menu; this menu displays a message telling you how many panels were copied and replaced.

Maintaining Library Definitions

Option **L** from the MODS : Panel Library Maintenance Menu displays the MODS : Library Definition Menu, as shown in Figure 16-3.

This menu can be used to define a library temporarily (for example, if you want to copy panels to or from a panels dataset defined to another system). If you require a permanantly available library, you should define it by including the appropriate commands in the NMINIT procedure (see the *SOLVE Management Services Implementation and Administration Guide* for details).

To make a library available for use, the dataset must be allocated to SOLVE, opened, and defined as a library. These steps can be performed using the ALLOC, UDBCTL OPEN, and LIBRARY commands respectively. The MODS : Library Definition Menu provides facilities for performing these steps, and for reversing them, without the need for you to issue any commands.

Figure 16-3. MODS : Library Definition Menu

```
SOLVPROD---------------- MODS : Library Definition Menu -----------------------
 Select Option ===>

    A   - Allocate, Open and Define Library
    O   - Open and Define Library
    D   - Define Library
    R   - Remove Library Definition
    C   - Remove Library Definition and Close
    U   - Remove Library Definition, Close and Unallocate
    X   - Exit

 Library Name ... ________

 File ID ........ ________ ( Optional A O D )

 DD Name ........ ________ ( Optional A O )

 Edit Allowed ... YES      ( Required A O D YES or NO, Default YES )

 Description .... DEFINED BY USER01_____________________________ ( Required A O D )

 Dataset Name ... ________________________________________________ ( Required A )

                            F9=Swap
```

The following input fields are shown on the MODS : Library Definition Menu. The fields you need to enter depend on the menu option you select.

Library Name

The name of the library to be defined or removed. This field is required for all options.

File Id

File Id is the name of the fileid to define on the UDBCTL OPEN command for option **A** and **O**, or the fileid of a previously opened file for option **D**. If omitted, **File Id** defaults to the name specified in the **Library Name** field (for options **A**, **O**, and **D**).

DD Name

The **DD** name which is used when the file is opened as a library. For option **A** this field provides a DDNAME to allocate, and for option **O** it provides the name of a previously allocated file to be opened. If omitted, **DD Name** defaults to the name specified in the **Library Name** field (for options **A** and **O**).

Edit Allowed

Specifies whether users are allowed to edit panels in this library (enter **YES**) or not (enter **NO**). This field is required for options **A**, **O** and **D**; if not specified, defaults to YES.

Description

An up to 40 character description of the library. If omitted, the **Description** defaults to DEFINED BY followed by your user ID, for options **A**, **O**, and **D**.

Dataset Name

Used by option **A** to define the name of the dataset being allocated. The **Dataset Name** must be a fully qualified dataset name without quotes and must be a VSAM KSDS.

The following options are available on the MODS : Library Definition Menu:

Option	Explanation
A	Allocate the specified dataset to SOLVE, open it for use and define it as a library
O	Open the already allocated file, and define it as a library
D	Define an already open file as a library
R	Remove the library definition from the system, close the file and deallocate the dataset from SOLVE
C	Remove the library definition and close the file
U	Remove the library definition

17

Maintaining the MODS Control File

This chapter discusses the facilities available for maintaining Managed Object Development Services (MODS) control files.

You can copy, move or delete definitions in MODS control files and can also compare two MODS control files to create a *difference file*, that can then be used to update another control file.

The facilities for control file maintenance allow you to:

- Copy data from one control file to another
  (for example, from TEST to PRODUCTION)
- Move data from one control file to another
- Delete data from a control file
- Compare two control files and create a difference file
- Use this difference file to make two control files identical
  (Apply the Differences)
- Browse a MODS definition of a control file as if browsing the definition through the MODS primary menu
- Search a control file for a string of characters
- List the entire contents of a control file
- Reset / Clear all MODS definitions from the MODS cache table

About Control Files

A MODS control file contains component definitions for use by NCL applications. It contains both application and common components as follows:

- Application dependent components:
  - $AR—Application Definitions
  - $HM—Help
  - $MH—Menus
  - $MS—Messages
  - $VM—Tables
  - $RW—Reports
  - $LH—Lists
  - $CR—Criteria
  - $PV—Panel Domains
  - $OS—Object Specifications
  - $LD—Language Services - Presentation Elements and String Definitions
- Common (non-application dependent) components:
  - $CM—Commands
  - $PS—Print Services Manager (PSM) definitions
  - $LD—Language Definitions

The components contained within a control file consist of sub components and entities as shown in Table 17-1.

You can choose to move, copy, delete or compare either:

- Entire applications (that is, all application dependent components for an application)
- Entire components within an application (for example, all menus for application $AB)
- All entities belonging to a subcomponent of a component (only applicable if subcomponents exist)
- Individual entities

Note

Subcomponents preceded with the ◆ character can be acted upon as individual entities, while those preceded with the ● character are acted upon automatically when the parent subcomponent is affected.

Warning

Due to the logical relationships between the records in a control file you *must not* perform maintenance using REPRO or other file utilities.

These utilities should not be used for any purpose other than to perform backups as they cannot take account of these logical relationships and could corrupt the control file with unpredictable consequences.

Table 17-1. MODS Components and Subcomponents/Entities

Component	Subcomponents / Entities
$HM - Help	◆ Application level help ◆ Function level help ◆ Window level help ◆Field level help
$MH - Menus	◆ Menu Definitions ● Menu Profiles
$MS - Messages	◆Message Definitions
$VM - Tables	◆ Table Definitions ● Table Entries
$RW - Reports	◆ Report Definitions ●Report Descriptions ●Report Headers ●Page Headers ●Control Break Headers ●Data Formats ●Control Break Trailers ●Page Trailers ●Report Trailers ●Sort Fields
$LH - Lists	◆List Definitions
$CR - Criteria	◆Criteria Definitions
$PV - Panel Domains	◆ Panel Domains ●Elements ● Paths
$AR - Application Register	◆Application Identifiers

Table 17-1. MODS Components and Subcomponents/Entities (Continued)

Component	Subcomponents / Entities
$OS - Object Services	◆ Object Control Record ●Classes ●Name Bindings ●Packages ●Package Attributes ●Package Actions ●Package Events ●Attributes ●Actions ●Events ●Parameters ●Data Types ●Relationships ●Data Domains ●Attribute Groups ●Attributes
$LD - Language Services Presentation Elements and String Definitions	◆Presentation Elements ◆String Definitions
$CM - Commands	◆Command Definitions
$PS - Print Services Manager definitions	◆ Defaults Definition ◆ Printer Definitions ◆ Form Definitions ◆ Setup Definitions ◆Default Printer Assignments
$LD - Language Definitions	◆Language Definitions

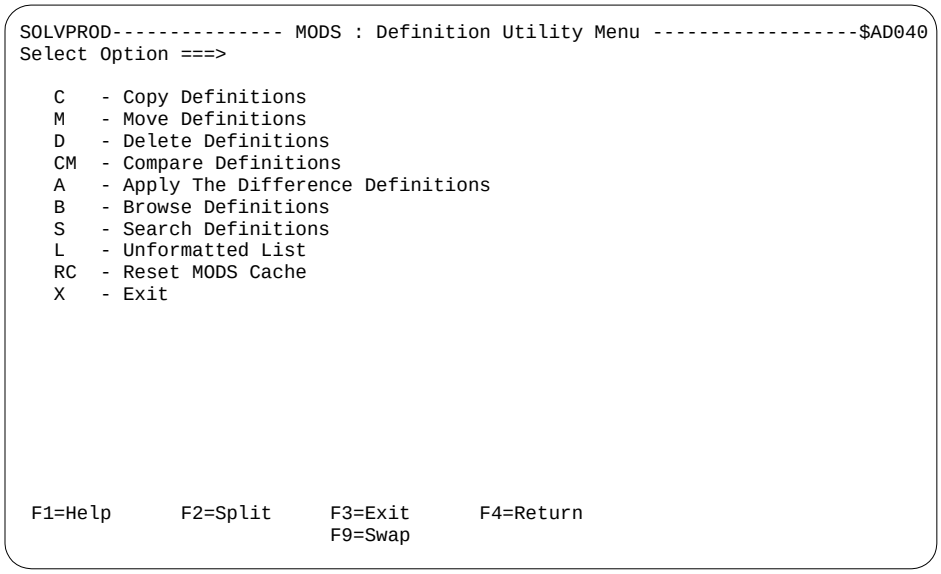
Accessing the Control File Maintenance Facilities

The control file maintenance facilities are accessed through the MODS : Definition Utility Menu, by selecting the following:

- Option **D** from the SOLVE : Primary Menu
- Option **AD** from the MODS : Primary Menu
- Option **U** from the MODS : Administration Menu

The MODS : Definition Utility Menu is illustrated in Figure 17-1.

Figure 17-1. MODS : Definition Utility Menu



The following options are available on the MODS : Definition Utility Menu:

Option	Explanation
C	Copy entities from one control file to another (see the section titled *Copying Components Between Control Files*, on page 17-8)
M	Move entities from one control file to another (see the section titled *Moving Components Between Control Files*, on page 17-18)
D	Delete entities from a control file (see the section titled *Deleting Components from the Control File*, on page 17-23)
CM	Compare two control files and create a difference file (see the section titled *Comparing Control Files*, on page 17-26)
A	Apply a difference file to a control file (see the section titled *Applying Differences*, on page 17-32)
B	Browse entities from a control file (see the section titled *Browsing the Control File*, on page 17-36)
S	Search a control file for a character string (see the section titled *Searching the Control File*, on page 17-38)
L	List all the data on a control file in summary form (see the section titled *Listing the Control File*, on page 17-40)
RC	Clear all MODS definitions from the MODS cache table (see the section titled *Resetting the MODS Cache*, on page 17-42)

Note

The MODS : Definition Utility Menu does not support the SOLVE Web file system. You cannot use these utilities to maintain any SOLVE Web files. For information on Web file utilities, see Appendix F, *SOLVE Web File Utilities*.

Panel navigation is illustrated in Figure 17-2.

Applying APAR and PUT Tapes

To apply a supplied PUT (Product Update Tape) or APAR (Authorized Program Analysis Report) tape, do the following:

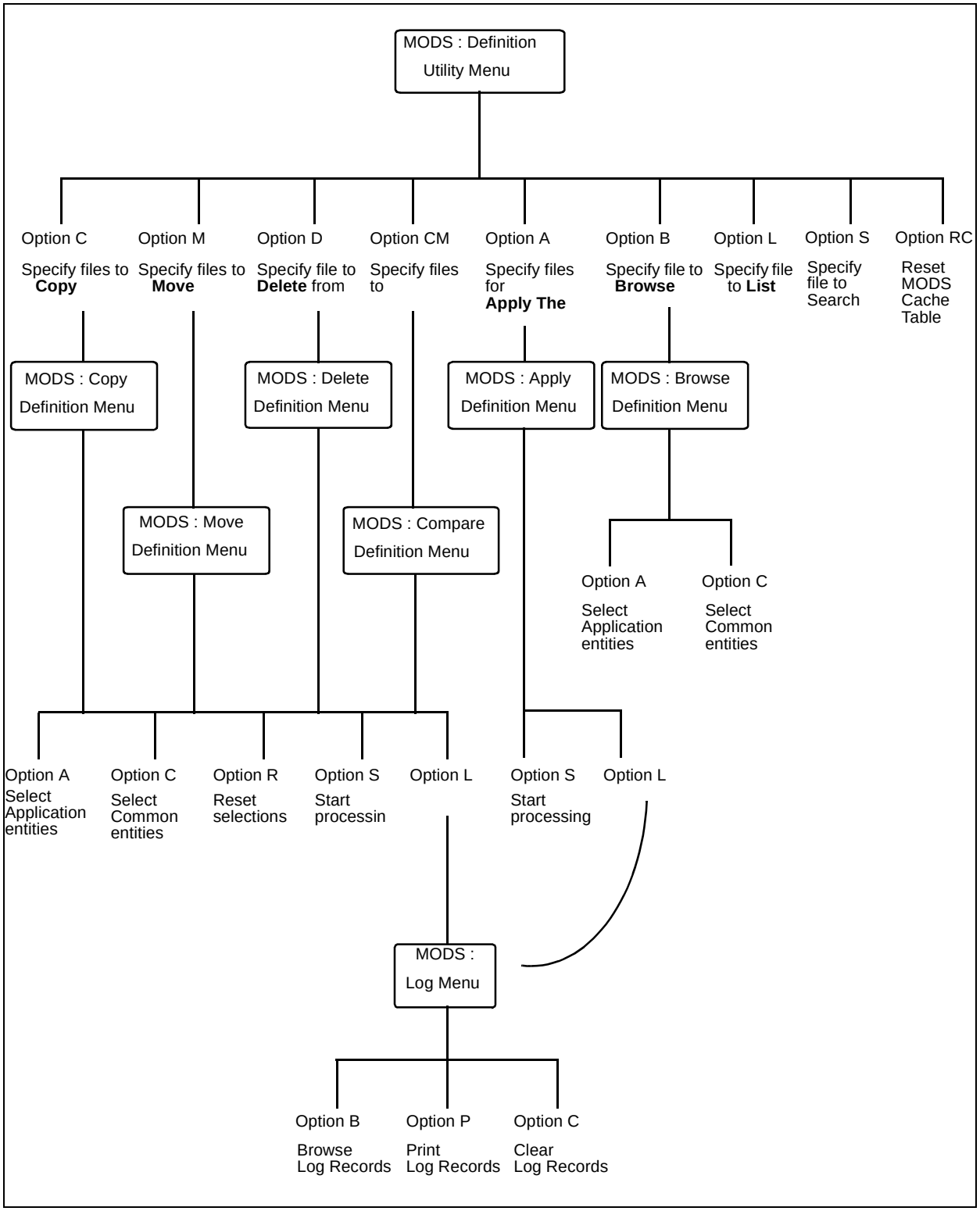
- Unload from the tape to a staging MODS control file.
- Apply this staging file to your existing MODS control file using option **A—Apply The Difference Definitions** from the MODS : Definition Utility Menu.

PUTs and APARs are created as difference files using the Compare option. The PUT or APAR Installation Instructions include information on the contents of the difference file. To display the contents of the file, use option **L—Unformatted List.** Individual records are tagged as Add, Update, or Delete.

Note

For applications prior to this release of the SOLVE product, use the Copy option instead of the Apply The Difference Definitions option.

Figure 17-2. Control File Maintenance—Panel Navigation



Copying Components Between Control Files

Option **C** from the MODS : Definition Utility Menu displays the MODS : Copy File Specification panel, as shown in Figure 17-3.

Use this panel to specify the control files to copy from and to, before using the MODS : Copy Definition Menu to select specific entities to be copied.

Figure 17-3. MODS : Copy File Specification panel

```
SOLVPROD----------------- MODS : Copy File Specification ----------------------
Command ===>

FROM: File ID ...........+ ________
      or Dataset Name .... ____________________________________________________
      Language Code ...... UK_

  TO: File ID ...........+ ________
      or Dataset Name .... ____________________________________________________

      Disp ............... SHR        (SHR/NEW/OLD)
      Volume ............. ______
      Prim & Sec Space ... 1__ , 1__  (cylinders)









 F1=Help      F2=Split     F3=Exit      F4=Return                F6=Action
                           F9=Swap
```

The fields displayed on the MODS : Copy File Specification panel are as follows:

FROM :

File ID
This (up to) 8 character name identifies the file to copy from.

Dataset Name
The name of the dataset to copy from.

Note

Either **File ID** or **Dataset Name** must be specified.

Language Code
The language code of the records that you want to copy and are present in the **FROM** file you have specified. This field has a default value of **UK**.

TO :

File ID
This name identifies the file to copy to.

Dataset Name
The name of the dataset to copy to.

Note

Either **File ID** or **Dataset Name** must be specified.

Disp
Specifies the disposition of the file—**SHR**, **OLD** or **NEW** (**NEW** applies only to new datasets). If **NEW**, a **Dataset Name** must be specified, not a **File ID**.

Volume
Specifies the volume to allocate a **NEW** dataset onto.

Prime & Sec Space
Specifies the primary and secondary storage to be allocated for a **NEW** dataset.

After completing this panel, press the ACTION key to display the MODS : Copy Definition Menu (Figure 17-4).

Use this menu to select the actual entities to be copied within the **FROM** file, and to initiate the copy process.

Figure 17-4. MODS : Copy Definition Menu

```
SOLVPROD---------------- MODS : Copy Definition Menu --------------------$AD050
Select Option ===>

   A   - Applications List
   C   - Common Services List
   R   - Reset Selections
   S   - Start Copy Process
   L   - Log Records
   X   - Exit

Appl ID Prefix .... ___  ( Optional A )
Replace Entity .... NO_  ( Required S , YES or NO )
Log ............... NO_  ( Required S , YES or NO )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the MODS : Copy Definition Menu are as follows:

Appl ID Prefix

Use this field to limit the list when you select option **A**. You can enter a string of up to three characters. All applications with IDs that start with the specified string of characters are listed. For example, if you enter Z, all applications with IDs that start with Z are listed.

Replace Entity

Specifies whether entities on the **TO** file are to be overwritten by entities of the same name from the **FROM** file:

- Enter **YES** to overwrite entities
- Enter **NO** to prevent entities of the same name from being copied

Log

Specifies whether copies performed are to be logged (enter **YES**) or not (enter **NO**).

The following options are available on the MODS : Copy Definition Menu:

Option	Explanation
A	Select Application entities to be copied (go to the MODS : Application List panel)
C	Select Common entities to be copied (go to the MODS : Component List panel)
R	Reset (clear) your selections
S	Start the copy process
L	Go to the MODS : Log Menu

A detailed description of these options follows.

Selecting Application Entities to Copy

Option **A** displays the MODS : Application List panel, as shown in Figure 17-5.

The top right corner displays the names of the **FROM** file (`MODSTST` in this example) and the **TO** file (`MODSUSR`).

Figure 17-5. MODS : Application List

```
SOLVPROD------------------- MODS : Application List ------------MODSTST=>MODSUSR
 Command ===>                                                   Scroll ===> PAGE

                     S=Select A=AllSelect XS=eXcludeSelect U=UnSelect B=Browse
    Appl  Description                        Msg Prefix                File ID
    $DD   MDS Data Dictionary                DD                        MODSDIS
    $DE   DEC SNA Interconnection            DE                        MODSDIS
    $DM   NDM API                            DM                        MODSUSR
    $EA   EASINET                            EA                        MODSDIS
    $EB   External Database Support          EB                        MODSDIS
    $ED   Edit Services/CAS Editor           ED                        MODSDIS
    $EI   External Interface Package         EI                        MODSDIS
    $ES   Expert System Foundation           ES                        MODSUSR
    $FK   CAS Function Key Handler           FK                        MODSDIS
    $FT   FTS                                FT                        MODSDIS
    $GM   Graphics Monitor                   GM                        MODSDIS
    $GP   CAS General Purpose Routines       GP                        MODSDIS
    $HD   NCS Help Desk                      HD                        MODSDIS
    $HM   CAS Help Manager                   HM                        MODSDIS
    $IM   INFO/MASTER                        IM                        MODSDIS
    $IN   Information Database               IN                        MODSDIS
    $IS   Information Services               IS                        MODSDIS
  F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
  F7=Backward  F8=Forward   F9=Swap                 F11=Right    F12=ResetSel
```

You can perform the following actions on any application(s) in the list, by using the appropriate mnemonic:

Mnemonic	Action
S	Go to the MODS : Component List panel, to select specific components of this application.
A	Select all entities in this application. This option is not available if entities have already been selected individually.
XS	Go to the MODS : Component List panel, to exclude specific components. This option is only available after an AllSelect has been performed on this application.
U	Undo all selections in this application. This option is available only if the Selected indicator is displayed.
B	Browse the selected application definition.

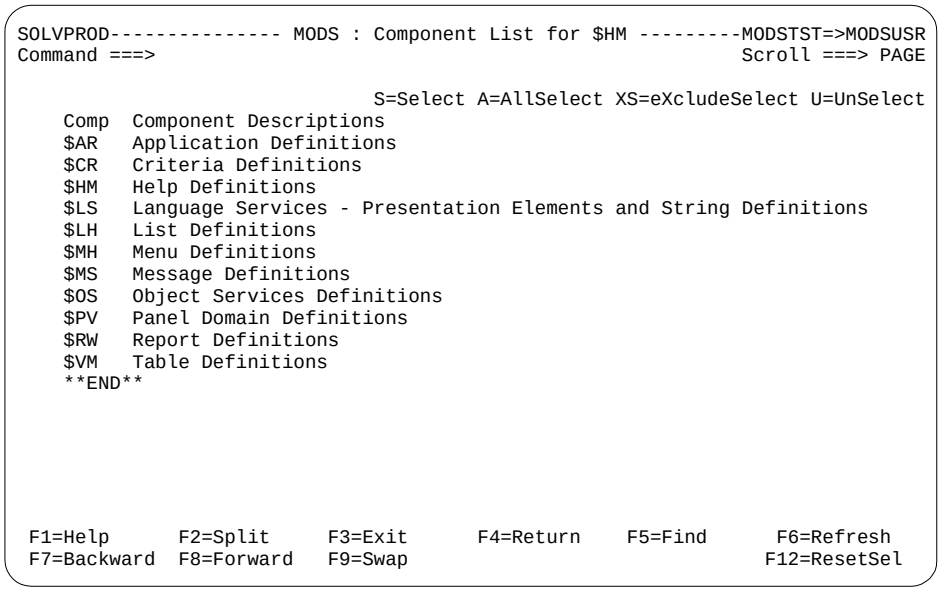
Note

The RESETSEL key clears *all* selections.

The **S** and **XS** mnemonics display the MODS : Component List panel, as shown in Figure 17-6.

The top right corner displays the names of the **FROM** file (`MODSTST` in this example) and the **TO** file (`MODSUSR`).

Figure 17-6. MODS : Component List (for Application Data)



Note

If a **FROM Language Code** other than **UK** is specified, the component list is restricted to those components that have national language support.

You can perform the following actions on any component(s) in the list, by using the appropriate mnemonic:

Mnemonic	Action
S	Display the appropriate Definition List (with an intermediate SubComponent List, where applicable), from which you can select individual entities.
A	Select all entities in this component. (This option is not available if entities have already been selected individually.)
XS	Display the appropriate Definition List (with an intermediate SubComponent List, where applicable), to exclude specific entities. This option is only available after an AllSelect has been performed on this component.
U	Undo all selections in this component. (This option is available only if the Selected indicator is displayed.)

Note

The RESETSEL key clears *all* selections.

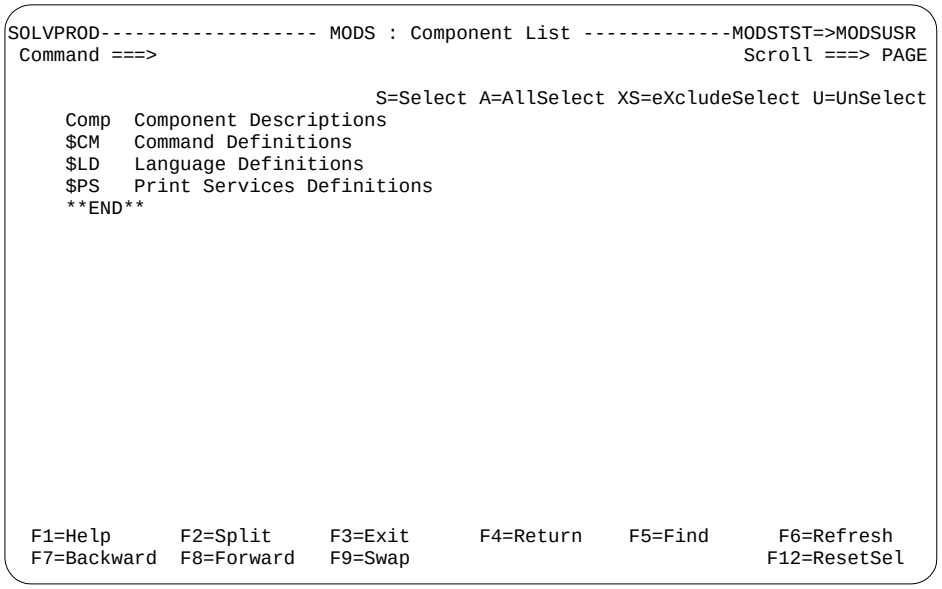
The individual definition lists display all entities for a component. You can perform the following actions on any entity in these lists, by using the appropriate mnemonic:

Mnemonic	Action
S	Select this entity
U	Undo selection of this entity
B	Browse this entity

Selecting Common Entities to Copy

Option **C** displays the MODS : Component List panel, as shown in Figure 17-7. The top right corner shows the names of the **FROM** file (`MODSTST` in this example) and the **TO** file (`MODSUSR`).

Figure 17-7. MODS : Component List (for Common Data)



Note

If you have specified a **FROM Language Code** other than **UK**, the common component list is unavailable as there are no Common Services that have national language support.

You can perform the following actions on any component(s) in the list, by using the appropriate mnemonic:

Mnemonic	Action
S	Display the appropriate Definition List (with an intermediate SubComponent List, where applicable), from which you can select individual entities.
A	Select all entities in this component. (This option is not available if entities have already been selected individually.)
XS	Display the appropriate Definition List (with an intermediate SubComponent List, where applicable), to exclude specific entities. This option is only available after an AllSelect has been performed on this component.
U	Undo all selections in this component. (This option is available only if the Selected indicator is displayed.)

Note

The RESETSEL key clears *all* selections.

The individual definition lists display all entities for a component. You can perform the following actions on any entity in these lists, by using the appropriate mnemonic:

Mnemonic	Action
S	Select this entity
U	Undo selection of this entity
B	Browse this entity

Resetting Selections

Option **R** clears all selections you have made, from both Application entities and Common entities.

You can also use the RESETSEL key, at any point, to clear your selections.

Copying Selected Data

After selecting all the required Application and/or Common entities, press the RETURN key (this takes you back to the MODS : Copy Definition Menu) and choose option **S**. This displays the MODS : Control File Entity Copy panel, with the message:

```
AD1306 ANALYSIS IN PROGRESS
```

This is followed by the message:

```
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO COPY
```

The MODS : Control File Entity Copy panel (Figure 17-8) summarizes the selected entities. The top right corner shows the names of the **FROM** file (MODSTST in this example) and the **TO** file (MODSUSR).

Figure 17-8. MODS : Control File Entity Copy Panel

```
SOLVPROD---------------- MODS : Control File Entity Copy ------MODSTST=>MODSUSR
Command ===>
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO COPY
   Copied  ToCopy  NoRep   ID  Cat  Description
                0      1  $LH  LD0  List Definitions
                2      0  $MH  000  Menu Definitions
                4      0  $MS  100  Message Definitions
                7      3  $OS  PC0  Object Services Dfns - Package Specs
                6      0  $OS  AT0  Object Services Dfns - Attribute Specs
 F1=Help      F2=Split                                          F6=Action
                           F9=Swap                            F12=Cancel
```

Press the ACTION key to copy the selected entities from the **FROM** file to the **TO** file. The **Copied** column tracks the copy processing, and the following message is displayed:

```
AD1311 COPY IN PROGRESS
```

The **NoRep** column lists entities that are not copied because the **Replace Entity** field on the MODS : Copy Definition Menu panel (Figure 17-4) is set to NO.

When the copy is complete, a final message is displayed:

```
AD1307 COPY COMPLETE
```

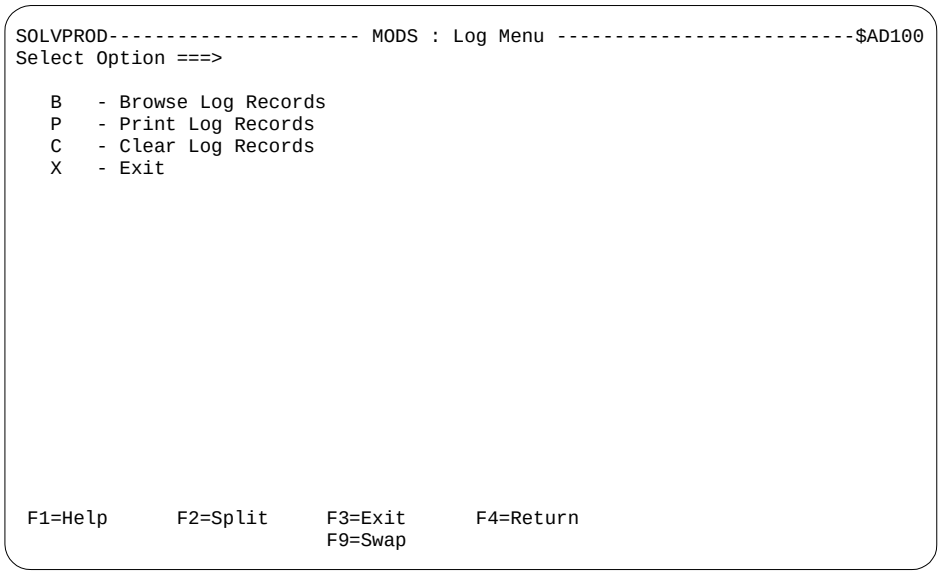
Note

If **Replace Entity** (on the MODS : Copy Definition Menu panel) is set to **NO**, entities on the **FROM** file which would overwrite entities on the **TO** file are not copied.

Accessing the Log

Option **L** displays the MODS : Log Menu, as illustrated in Figure 17-9.

Figure 17-9. MODS : Log Menu



Note

If **Log** was set to **NO**, the following message is displayed when you access the MODS : Log Menu:

```
AD2009 LOG TABLE NOT FOUND
```

The following options are available on the MODS : Log Menu:

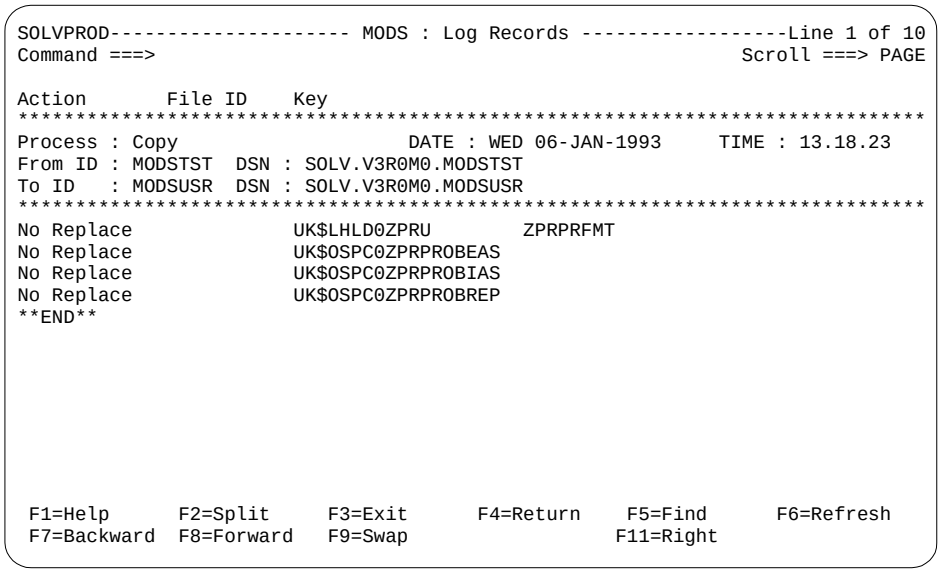
Option	Explanation
B	Browse the Log
P	Print the Log
C	Clear the Log

A detailed description of these options follows.

Browsing Log Records

Option **B** displays the MODS : Log Records panel, as shown in Figure 17-10.

Figure 17-10. MODS : Log Records Panel



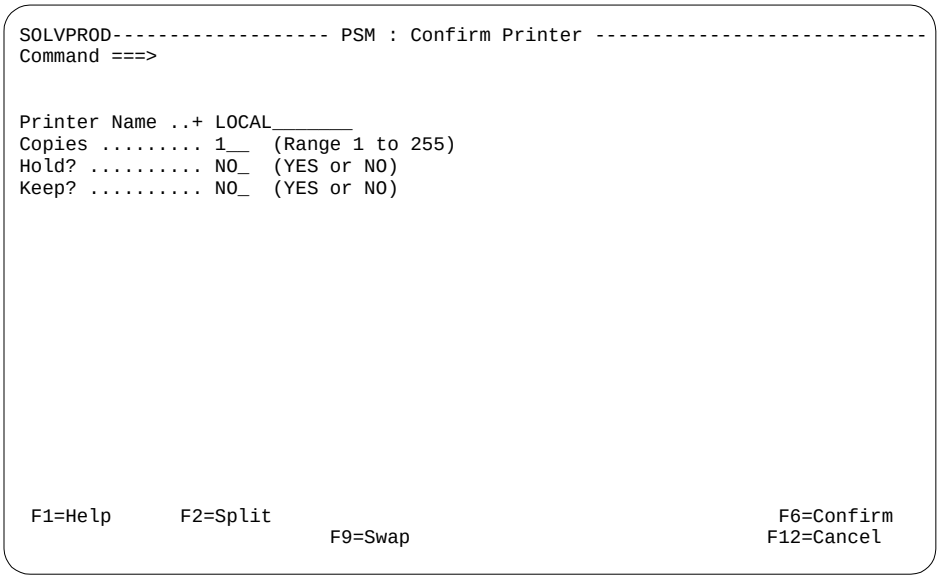
The number in the top right corner indicates the number of records in the log.

A header record is displayed at the beginning of each process in the log which shows the **FROM** and **TO** MODS control files and the name of the dataset for each as well as the time and date that the process was initiated.

Printing Log Records

Option **P** displays the PSM : Confirm Printer panel, as shown in Figure 17-11.

Figure 17-11. PSM : Confirm Printer panel



The fields displayed on the PSM : Confirm Printer panel are as follows:

Printer Name
The name of the printer which is to print the log. (The printer must be defined to PSM—for details, see the chapter titled *Print Services Manager (PSM)* in the *SOLVE Management Services User's Guide.*)

Copies
The number of copies to print. Enter a number between 1 and 255.

Hold?
Determines whether or not the print request is to be held on the spool (enter **YES** or **NO**). If the request is held, the log is not printed until the request is released.

Keep?
Determines whether or not the print request is to be deleted from the spool after printing. Enter **YES** to retain, **NO** to delete.

Press the CONFIRM key to print the log or the CANCEL key to cancel printing.

Clearing Log Records

Option **C** deletes the log records, displaying the following message:

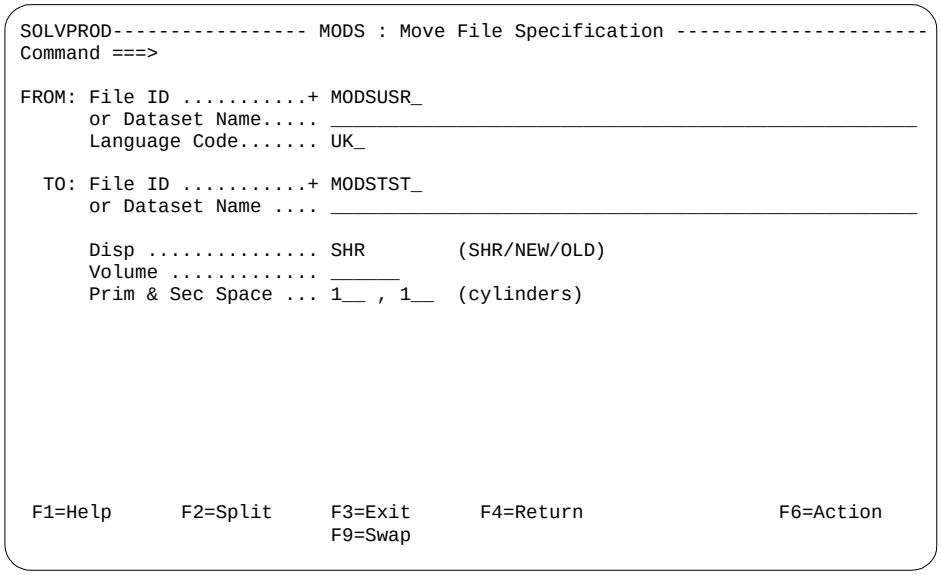
```
AD0704 TABLE CLEARED
```

Moving Components Between Control Files

Option **M** from the MODS : Definition Utility Menu displays the MODS : Move File Specification panel, as shown in Figure 17-12.

Use this panel to specify the control files to move entities from and to, before using the MODS : Move Definition Menu to select specific entities to be moved.

Figure 17-12. MODS : Move File Specification Panel



The fields displayed on the MODS : Move File Specification panel are as follows:

FROM :

File ID
The name of the file to move entities from.

Dataset Name
The name of the dataset to move entities from.

Note

Either **File ID** or **Dataset Name** must be specified.

Language Code
The language code of the records that you want to move and are present in the **FROM** file you have specified. This field has a default value of **UK**.

TO :

File ID
The name of the file to move entities to.

Dataset Name
The name of the dataset to move entities to.

Note

Either **File ID** or **Dataset Name** must be specified.

Disp

Specifies the disposition of the file—SHR**,** OLD or NEW (NEW applies only to new datasets). If NEW, a **Dataset Name** must be specified, not a **File ID**.

Volume

Specifies the volume to allocate a NEW dataset onto.

Prime & Sec Space

Specifies the primary and secondary storage to be allocated for a NEW dataset.

After completing this panel, press the ACTION key to display the MODS : Move Definition Menu (Figure 17-13).

Use this menu to select the actual entities to be moved from the **FROM** file, and to initiate the move process.

Figure 17-13. MODS : Move Definition Menu

```
SOLVPROD---------------- MODS : Move Definition Menu --------------------$AD090
Select Option ===>

   A   - Applications List
   C   - Common Services List
   R   - Reset Selections
   S   - Start Move Process
   L   - Log Records
   X   - Exit

Appl ID Prefix .... ___  ( Optional A )
Replace Entity .... NO_  ( Required S , YES or NO )
Log ............... NO_  ( Required S , YES or NO )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the MODS : Move Definition Menu are as follows:

Appl ID Prefix

Use this field to limit the list when you select option **A**. You can enter a string of up to three characters. All applications with IDs that start with the specified string of characters are listed. For example, if you enter Z, all applications with IDs that start with Z are listed.

Replace Entity

Specifies whether entities on the **TO** file are to be overwritten by entities of the same name from the **FROM** file:

- Enter **YES** to overwrite entities
- Enter **NO** to prevent entities of the same name from being moved

Log

Specifies whether moves performed are to be logged (enter **YES**) or not (enter **NO**).

The following options are available on the MODS : Move Definition Menu:

Option	Explanation
A	Select Application entities to be moved (go to the MODS : Application List panel)
C	Select Common entities to be moved (go to the MODS : Component List panel)
R	Reset (clear) your selections
S	Start the move process
L	Go to the MODS : Log Menu

A detailed description of these options follows.

Selecting Application Entities to Move

Option **A** displays the MODS : Application List panel, as shown in Figure 17-5 on page 17-11.

You can select entities as described in the section titled *Selecting Application Entities to Copy*, on page 17-10.

Selecting Common Entities to Move

Option **C** displays the MODS : Component List panel, as shown in Figure 17-7 on page 17-13.

You can select entities as described in the section titled *Selecting Common Entities to Copy*, on page 17-13.

Resetting Selections

Option **R** clears all the selections you have made, from both Application entities and Common entities. (Note that you can also use the RESETSEL key, at any point, to clear your selections.)

Moving Selected Data

After selecting all the required Application and/or Common entities press the RETURN key and then choose option **S** from the MODS : Move Definition Menu (which is redisplayed).

This displays the MODS : Control File Entity Move panel, with the message:

```
AD1306 ANALYSIS IN PROGRESS
```

Followed by the message:

```
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO MOVE
```

The MODS : Control File Entity Move panel (Figure 17-14) summarizes the selected entities.

Figure 17-14. MODS : Control File Entity Move panel

```
SOLVPROD---------------- MODS : Control File Entity Move ------MODSUSR=>MODSTST
Command ===>
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO MOVE
    Moved  ToMove  NoRep   ID  Cat  Description
                60      0  $CM  CT0  Command Definitions
                 1      0  $PS  DD0  Print Services - Defaults
                17      0  $PS  PD0  Print Services - Printer Definitions
                12      0  $PS  FD0  Print Services - Form Definitions
                 9      0  $PS  SD0  Print Services - Setup Definitions
                97      0  $PS  DA0  Print Services - Default Ptr Assignment
 F1=Help      F2=Split                                          F6=Action
                           F9=Swap                            F12=Cancel
```

Press the ACTION key to move the selected entities from the **FROM** file to the **TO** file. The **Moved** column tracks the move processing, and the following message is displayed:

```
AD1311 MOVE IN PROGRESS
```

The NoRep column lists entities that are not moved because the **Replace Entity** field on the MODS : Move Definition Menu panel (Figure 17-14) was set to NO.

When all the entities have been moved, a final message is displayed:

```
AD1307 MOVE COMPLETE
```

Accessing the Log

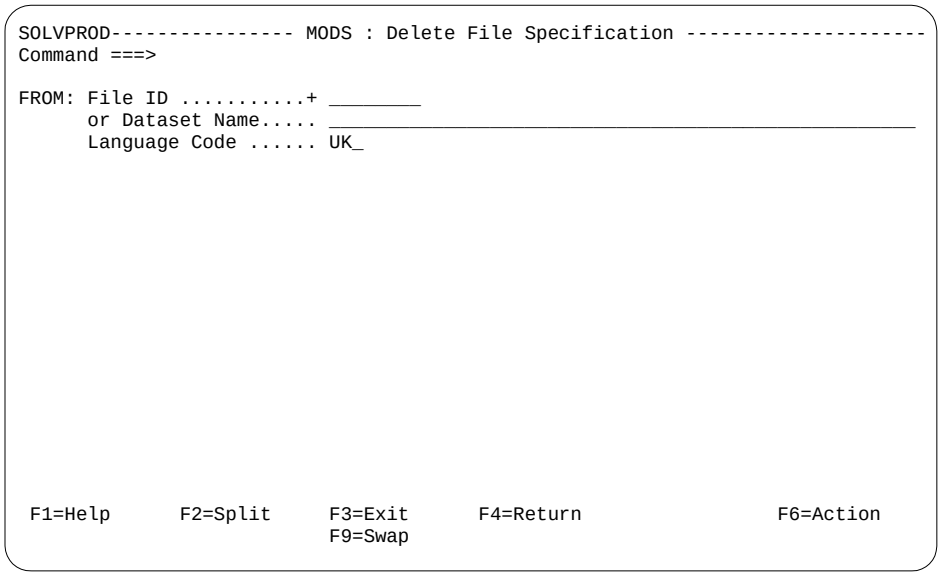
Option **L** displays the MODS : Log Menu, as illustrated in Figure 17-9 on page 17-16.

Deleting Components from the Control File

Option **D** from the MODS : Definition Utility Menu displays the MODS : Delete File Specification panel, as shown in Figure 17-15.

This panel requires you to specify the control file to delete from, before using the MODS : Delete Definition Menu to select specific entities to be deleted.

Figure 17-15. MODS : Delete File Specification Panel



The fields displayed on the MODS : Delete File Specification panel are as follows:

FROM :

File ID
This name identifies the file to delete from.

Dataset Name
The name of the dataset to delete from.

Note

Either **File ID** or **Dataset Name** must be specified.

Language Code

The language code of the records that you want to delete and are present in the **FROM** file you have specified. This field has a default value of **UK**.

After completing this panel, press the ACTION key to display the MODS : Delete Definition Menu (Figure 17-16). Use this menu to select the actual entities to be deleted, and to initiate the delete process.

Figure 17-16. MODS : Delete Definition Menu

```
SOLVPROD--------------- MODS : Delete Definition Menu ------------------$AD060
Select Option ===>

   A   - Applications List
   C   - Common Services List
   R   - Reset Selections
   S   - Start Delete Process
   L   - Log Records
   X   - Exit

Appl ID Prefix .... ___  ( Optional A )
Log ............... NO_  ( Required S , YES or NO )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the MODS : Delete Definition Menu are as follows:

Appl ID Prefix

Use this field to limit the list when you select option **A**. You can enter a string of up to three characters. All applications with IDs that start with the specified string of characters are listed. For example, if you enter Z, all applications with IDs that start with Z are listed.

Log

Specifies whether deletions are to be logged (enter **YES**) or not (enter **NO**).

The following options are available on the MODS : Delete Definition Menu:

Option	Explanation
A	Select Application entities to be deleted (go to the MODS : Application List panel)
C	Select Common entities to be deleted (go to the MODS : Component List panel)
R	Reset (clear) your selections
S	Start the delete process
L	Go to the MODS : Log Menu (see Figure 17-9 on page 17-16)

A detailed description of these options follows.

Selecting Application Entities to Delete

Option **A** displays the MODS : Application List panel, as shown in Figure 17-5 on page 17-11. You can select entities as described in the section titled *Selecting Application Entities to Copy*, on page 17-10.

Selecting Common Entities to Delete

Option **C** displays the MODS : Component List panel, as shown in Figure 17-7 on page 17-13. You can select entities as described in the section titled *Selecting Common Entities to Copy*, on page 17-13.

Resetting Selections

Option **R** clears all the selections you have made, from both Application entities and Common entities. (Note that you can also use the RESETSEL key, at any point, to clear your selections.)

Deleting Selected Data

After selecting all the required Application and/or Common entities, choose option **S** from the MODS : Delete Definition Menu. This displays the MODS : Control File Entity Delete panel, with the message:

```
AD1306 ANALYSIS IN PROGRESS
```

This is followed by the message:

```
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO DELETE
```

The MODS : Control File Entity Delete panel (Figure 17-17) summarizes the selected entities.

Figure 17-17. MODS : Control File Entity Delete Panel

```
SOLVPROD---------------- MODS : Control File Entity Delete --------DEL=>MODSUSR
Command ===>
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO DELETE
  Deleted  ToDelete   ID  Cat  Description
                60      0  $CM  CT0  Command Definitions
                 1      0  $PS  DD0  Print Services - Defaults
                17      0  $PS  PD0  Print Services - Printer Definitions
                12      0  $PS  FD0  Print Services - Form Definitions
                 9      0  $PS  SD0  Print Services - Setup Definitions
                97      0  $PS  DA0  Print Services - Default Ptr Assignment
 F1=Help      F2=Split                                          F6=Action
                           F9=Swap                            F12=Cancel
```

Press the ACTION key to delete the selected entities from the specified file. The **Deleted** column tracks the delete processing, and the following message is displayed:

```
AD1311 DELETE IN PROGRESS
```

When the deletion is complete, a final message is displayed:

```
AD1307 DELETION COMPLETE
```

Accessing the Log

Option **L** displays the MODS : Log Menu, as illustrated in Figure 17-9 on page 17-16.

Comparing Control Files

This section describes how to compare two control files in order to determine where they differ. These options are used to ensure that control files are consistent. You can examine differences between the files and then apply found differences to the target file.

To compare two control files, select option **CM** from the MODS : Definition Utility Menu. This displays the MODS : Compare File Specification panel, as shown in Figure 17-18.

This panel requires you to specify the two control files to compare (NEW and OLD), plus a file to write the differences to (DIFF). You can then use the MODS : Compare Menu to select specific entities from the new file to be compared to the old file.

Figure 17-18. MODS : Compare File Specification Panel

```
SOLVPROD--------------- MODS : Compare File Specification ---------------------
Command ===>

 NEW: File ID 1 .........+ ________
      or Dataset Name..... ____________________________________________________
      Language Code ...... UK_

 OLD: File ID 2 .........+ ________
      or Dataset Name..... ____________________________________________________
      Language Code ...... UK_

DIFF: File ID ............ ________
      or Dataset Name .... ____________________________________________________

      Disp ............... OLD        (SHR/NEW/OLD)
      Volume ............. ______
      Prim & Sec Space ... 1__ , 1__  (cylinders)
      Reset .............. YES        (YES/NO - Clear Existing Data)



 F1=Help      F2=Split     F3=Exit      F4=Return                F6=Action
                           F9=Swap
```

The fields displayed on the MODS : Compare File Specification panel are as follows:

NEW :

File ID
The (up to) 8 character name of the new file to compare.

Dataset Name
The name of the new dataset to compare.

Note

Either **File ID** or **Dataset Name** must be specified.

Language Code
The language code of the modified records. Any records written to the Difference file have this language code. This field has a default value of **UK**.

OLD :

File ID
The name of the old file to compare.

Dataset Name
The name of the old dataset to compare.

Note
Either **File ID** or **Dataset Name** must be specified.

Language Code
The language code of the OLD records. This field has a default value of **UK**.

DIFF :

File ID
The name of the file to write the differences to.

Dataset Name
The name of the dataset to write the differences to.

Note
Either ***File ID*** or ***Dataset Name*** must be specified.

Disp
Specifies the disposition of the difference file—SHR**,** OLD or NEW (NEW applies only to new datasets).

Volume
Specifies the volume onto which to allocate the difference file.

Prim & Sec Space
Specifies the primary and secondary storage to be allocated for the difference file.

Reset
Specifies whether existing data on the difference file is to be overwritten by the compare results:

- Enter **YES** to overwrite the difference file (this resets the file before processing)
- Enter **NO** to leave existing data intact

Note
If **Disp** is set to NEW, **Reset** must be NO.

After completing this panel, press the ACTION key to go to the MODS : Compare Definitions Menu (Figure 17-19).

Use this menu to select the actual entities to be compared, from the new file, and to initiate the compare process.

Figure 17-19. MODS : Compare Definitions Menu

```
SOLVPROD-------------- MODS : Compare Definitions Menu ------------------$AD070
Select Option ===>

   A   - Applications List
   C   - Common Services List
   R   - Reset Selections
   S   - Start Compare Process
   L   - Log Records
   X   - Exit

Appl ID Prefix .... ___  ( Optional A )
Log ............... NO_  ( Required S , YES or NO )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the MODS : Compare Definitions Menu are as follows:

Appl ID Prefix

Use this field to limit the list when you select option **A**. You can enter a string of up to three characters. All applications with IDs that start with the specified string of characters are listed. For example, if you enter Z, all applications with IDs that start with Z are listed.

Log

Specifies whether the compare action is to be logged (enter **YES**) or not (enter **NO**).

The following options are available on the MODS : Compare Definitions Menu:

Option	Explanation
A	Select Application entities to be compared (go to the MODS : Application List panel)
C	Select Common entities to be compared (go to the MODS : Component List panel)
R	Reset (clear) your selections
S	Start the comparison process
L	Go to the MODS : Log Menu (see Figure 17-9 on page 17-16)

A detailed description of these options follows.

Selecting Application Entities to Compare

Option **A** displays the MODS : Application List panel, as shown in Figure 17-5 on page 17-11.

You can select entities as described in the section titled *Selecting Application Entities to Copy*, on page 17-10.

Selecting Common Entities to Compare

Option **C** displays the MODS : Component List panel, as shown in Figure 17-7 on page 17-13.

You can select entities as described in the section titled *Selecting Common Entities to Copy*, on page 17-13.

Resetting Selections

Option **R** clears all selections you have made, from both Application entities and Common entities. (Note that you can also use the RESETSEL key, at any point, to clear your selections.)

Comparing Selected Data

After selecting all the required Application and/or Common entities, choose option **S** from the MODS : Compare Definitions Menu. This displays the MODS : Control File Entity Compare panel, with the message:

```
AD1306 ANALYSIS IN PROGRESS
```

This is followed by the message:

```
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO GENERATE DIFFERENCE
RECORDS
```

The MODS : Control File Entity Compare panel (Figure 17-20) summarizes the selected entities.

Figure 17-20. MODS : Control File Entity Compare Panel

```
SOLVPROD---------------- MODS : Control File Entity Compare ---MODSUSR=>MODSTST
Command ===>
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO GENERATE DIFFERENCE RECORDS
   Copied   Upd  Add  Del   ID  Cat  Description
              6    4    0  $CR  CD0  Criteria Definitions
              0    1    0  $MH  000  Menu Definitions









 F1=Help      F2=Split                                          F6=Action
                          F9=Swap                             F12=Cancel
```

Press the ACTION key to run the comparison between the NEW file and the OLD file. The **Copied** column tracks the comparison processing, and the following message is displayed:

```
AD1311 DIFFERENCE RECORD GENERATION IN PROGRESS
```

When the comparison is complete, a final message is displayed:

```
AD1307 DIFFERENCE RECORD GENERATION COMPLETE
```

Every difference found results in the entire entity being written to the DIFF file. The entities are flagged as follows:

- Entities that exist on the NEW file but not on the OLD file are flagged as ADDED.
- Entities that exist on the OLD file but not on the NEW file are flagged as DELETED.
- Entities that are different on the two files are flagged as UPDATED.

Accessing the Log

Option **L** displays the MODS : Log Menu, as illustrated in Figure 17-9 on page 17-16.

Applying Differences

Option **A** from the MODS : Utility Menu displays the MODS : Apply File Specification panel, as shown in Figure 17-21. This panel requires you to specify:

- **The FROM file**—the difference file generated by running a comparison between two control files (the DIFF file as described in the section *Comparing Control Files*, on page 17-26).
- **The TO file**—the file which is to be updated according to the differences recorded in the FROM file. (The TO file is the OLD file as described in the section *Comparing Control Files*, on page 17-26).

Figure 17-21. MODS : Apply File Specification Panel

```
SOLVPROD---------------- MODS : Apply File Specification ----------------------
Command ===>

FROM: File ID ............ ________   ( DIFF File from Comparison )
      or Dataset Name..... ____________________________________________________
      Language Code ...... UK_

  TO: File ID ...........+ ________
      or Dataset Name .... ____________________________________________________
      Language Code ...... UK_









 F1=Help      F2=Split     F3=Exit      F4=Return                F6=Action
                           F9=Swap
```

The fields displayed on the MODS : Apply File Specification panel are as follows:

FROM :

File ID
The name of the file in which the differences are recorded.

Dataset Name

The name of the dataset in which the differences are recorded.

Note

Either **File ID** or **Dataset Name** must be specified.

Language Code

The language code of the records present on the Difference file. This field has a default value of **UK**.

TO :

File ID

The name of the file to be updated.

Dataset Name

The name of the dataset to be updated.

Note

Either **File ID** or **Dataset Name** must be specified.

Language Code

The language code of the records to be added, updated, or deleted that are in the **TO** file you have specified. The value in this field can be different to the **FROM Language Code**. This field has a default value of **UK**.

After completing this panel, press the ACTION key to go to the MODS : Apply Definition Menu (Figure 17-22), from which you can initiate the apply process.

Figure 17-22. MODS : Apply Definition Menu

```
SOLVPROD---------------- MODS : Apply Definition Menu ------------------$AD110
Select Option ===>

   S   - Start Apply Process
   L   - Log Records
   X   - Exit

Replace Entity .... YES
Log ............... NO_  ( Required S , YES or NO )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The fields displayed on the MODS : Apply Definition Menu are as follows:

Replace Entity
Specifies whether entities on the TO file are to be overwritten by entities of the same name from the FROM file:

- Enter YES to overwrite entities
- Enter NO to prevent entities of the same name from being copied

Log
Specifies whether the apply action is to be logged (enter **YES**) or not (enter **NO**).

The following options are available on the MODS : Apply Definition Menu:

Option	Explanation
S	Start the apply process
L	Go to the MODS : Log Menu (see Figure 17-9 on page 17-16)

A detailed description of these options follows.

Applying the Differences

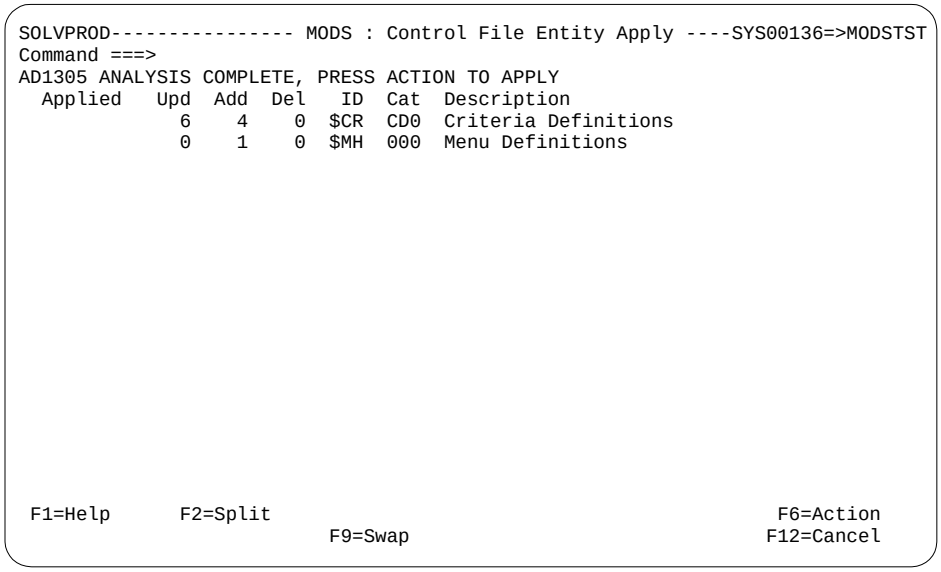
Choose option **S** to initiate the apply-the-differences processing. This option displays the MODS : Control File Entity Apply panel, with the message:

```
AD1306 ANALYSIS IN PROGRESS
```

Followed by the message:

```
AD1305 ANALYSIS COMPLETE, PRESS ACTION TO APPLY
```

Figure 17-23. MODS : Control File Entity Apply Panel



The MODS : Control File Entity Apply panel (Figure 17-23) summarizes the processing to be performed:

- The **Upd** column displays the number of entities on the TO file to be updated (overwritten).
- The **Add** column displays the number of entities to be added to the TO file.
- The **Del** column displays the number of entities to be deleted from the TO file.

Press the ACTION key to apply the differences recorded on the FROM file to the TO file. The **Applied** column tracks the processing, and the following message is displayed:

```
AD1311 APPLY IN PROGRESS
```

When processing is complete, a final message is displayed:

```
AD1307 APPLY COMPLETE
```

Accessing the Log

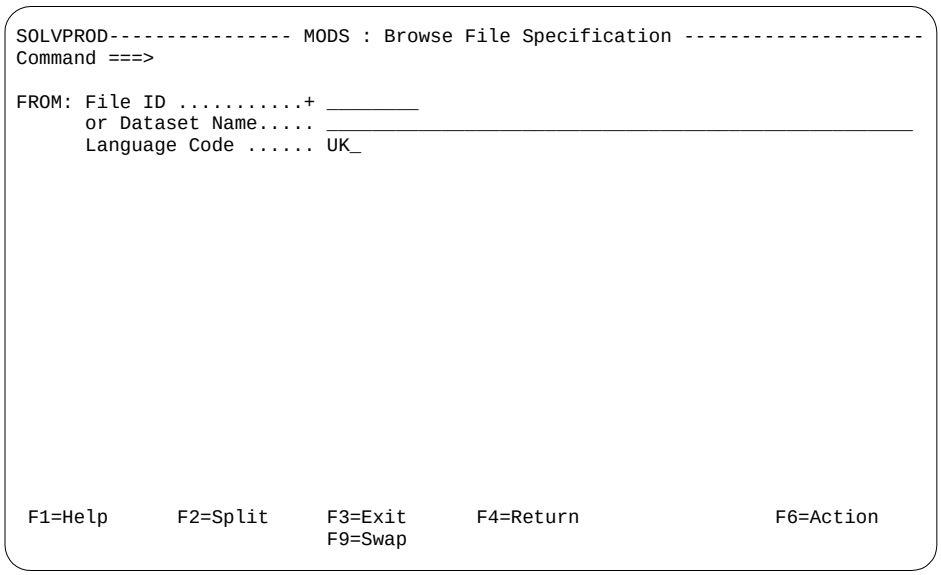
Option **L** displays the MODS : Log Menu, as illustrated in Figure 17-9 on page 17-16.

Browsing the Control File

Option **B** from the MODS : Definition Utility Menu displays the MODS : Browse File Specification panel, as shown in Figure 17-24.

This panel requires you to specify the control file from which to browse entities, before using the MODS : Browse Definition Menu to select specific entities to browse.

Figure 17-24. MODS : Browse File Specification Panel



The fields displayed on the MODS : Browse File Specification panel are as follows:

FROM :

File ID
The name of the file to browse.

Dataset Name
The name of the dataset to browse.

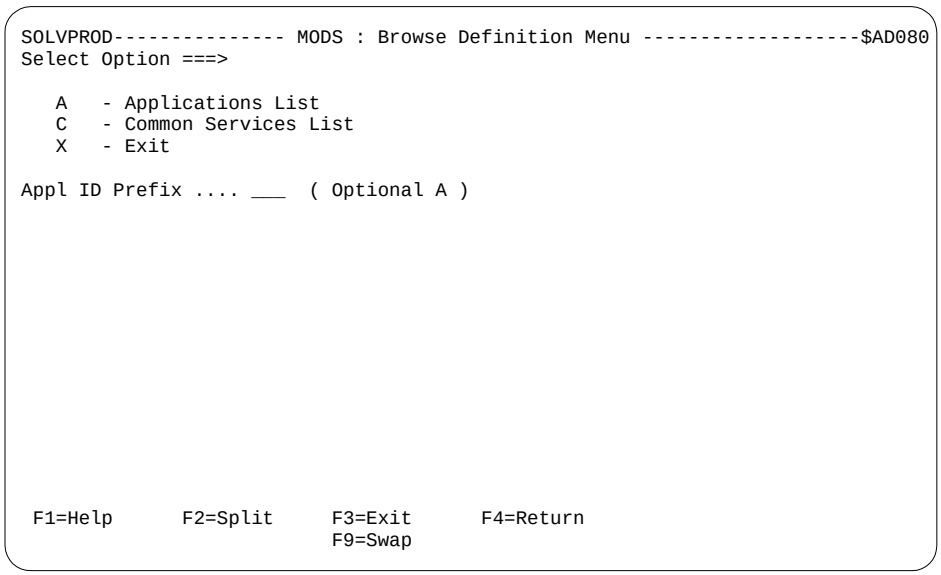
Note

Either **File ID** or **Dataset Name** must be specified.

Language Code
The language code of the records that you want to browse and are present in the **FROM** file you have specified. This field has a default value of **UK**.

After completing this panel, press the ACTION key to go to the MODS : Browse Definition Menu (Figure 17-25). Use this menu to select the actual entities to browse.

Figure 17-25. MODS : Browse Definition Menu



The field displayed on the MODS : Browse Definition Menu is as follows:

Appl ID Prefix

Use this field to limit the list when you select option **A**. You can enter a string of up to three characters. All applications with IDs that start with the specified string of characters are listed. For example, if you enter Z, all applications with IDs that start with Z are listed.

The following options are available on the MODS : Browse Definition Menu:

Option	Explanation
A	Select Application entities to browse
C	Select Common entities to browse

A detailed description of these options follows.

Browsing Application Entities

Option **A** displays the MODS : Application List panel, as shown in Figure 17-5 on page 17-11.

You can select entities to browse as described in the section titled *Selecting Application Entities to Copy,* on page 17-10.

Browsing Common Entities

Option **C** displays the MODS : Component List panel, as shown in Figure 17-7 on page 17-13.

You can select entities to browse as described in the section titled *Selecting Common Entities to Copy*, on page 17-13.

Searching the Control File

Option **S** from the MODS : Definition Utility Menu displays the MODS : Search Definitions panel, as shown in Figure 17-26.

This panel requires you to specify the control file to be searched, and the character string to be searched for. Each instance of the string found by the search is listed in a PSM report.

Figure 17-26. MODS : Search Definition Panel

```
SOLVPROD--------------- MODS : Search Definitions -----------------------------
Command ===>

Appl ID ..................+ ___
Component ................+ ___
Search String ..............  ________________________________________________
File ID ..................+ ________
Exclude Help and Messages?  YES
Language Code .............. UK_









 F1=Help      F2=Split     F3=Exit                              F6=Action
                           F9=Swap
```

The fields displayed on the MODS : Search Definitions panel are as follows:

Appl ID

Enter the 3-character identifier of the application whose entities are to be searched.

Component

Enter the 3-character category of the component whose entities are to be searched.

Search String

Enter the string to be searched for.

File ID

Enter the (up to) 8-character name of the file to be searched.

Exclude Help and Messages?

Provides the option to exclude the help category ($HM) and the message category ($MS) from the search. This field is ignored if a **Category** has been specified, as only records in a specified category are searched. This field has a default value of YES.

Language Code

The language code of the records that you want to search. This field has a default value of **UK**.

The search results in a PSM report that lists MODS records (if found) containing the specified string of characters. To start the search, press the ACTION key. The PSM : Confirm Printer panel appears.

If necessary, change the values in the fields, then press the CONFIRM key to start the search. If the number of records to be searched is 100 or more, a panel appears on your screen to advise you of the progress of the search. When the search is complete, a message appears on your screen to advise you the success of failure of the search.

If the report is on hold, you can use the **PQ**[UEUE] command to access the PSM output queue and view the report.

Considerations

When you study your search results, bear in mind the following:

- The search result lists only the first instance of a found string in a record.
- Record keys longer than 34 characters are truncated.
- During the search, a MODS record is accessed in chunks. The search is performed on each chunk and *not* across chunks: that is, if a string straddles two chunks, the string is *not* found. Shortening the search string can minimize this problem.

Listing the Control File

Option **L** from the MODS : Definition Utility Menu displays the MODS : Unformatted List of File Specification panel, as shown in Figure 17-27. This panel requires you to specify the control file to list.

Figure 17-27. MODS : Unformatted List of File Specification Panel

```
SOLVPROD-------- MODS : Unformatted List of File Specification ---------------
Command ===>

FROM: File ID ...........+ MODSUSR
      or Dataset Name..... _____________________________________________________









 F1=Help      F2=Split     F3=Exit      F4=Return                F6=Action
                           F9=Swap
```

The fields displayed on the MODS : Unformatted List of File Specification panel are as follows:

FROM :

File ID
The name of the file to list.

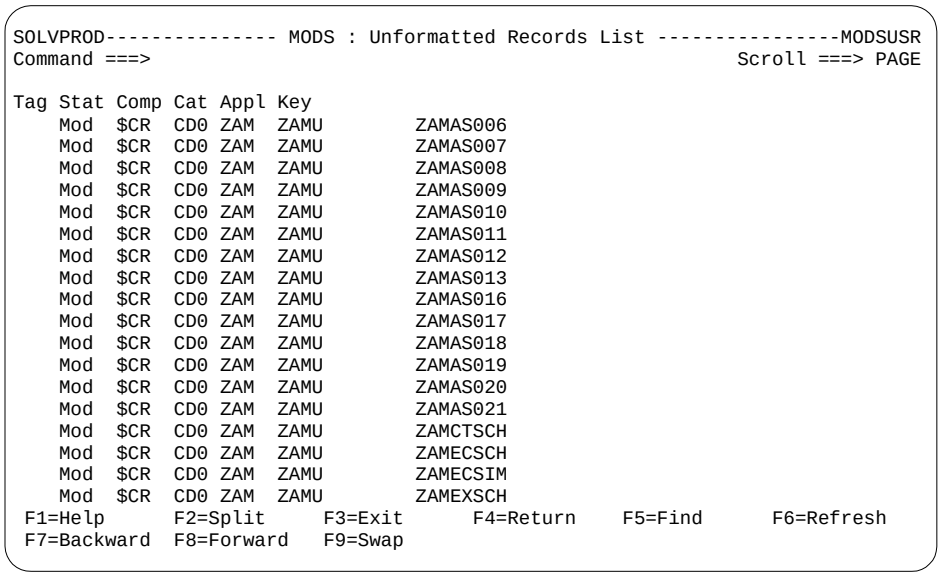
Dataset Name
The name of the dataset to list.

Note

Either **File ID** or **Dataset Name** must be specified.

After completing this panel, press the ACTION key to display the MODS : Unformatted Records List panel (Figure 17-28). This list summarizes the contents of the control file.

Figure 17-28. MODS : Unformatted Records List



The fields displayed on the MODS : Unformatted Records List panel are as follows:

Tag

Is only applicable to a difference file and is one of the following:

- Add—adds the record when the difference file is applied.
- Del—deletes the record when the difference file is applied.
- Upd—updates any existing record when the difference file is applied.

Stat

Is the status of the record as follows:

- Mod—indicates that the record has been modified.
- VRM—is the version, release, and modification levels set at the creation of the control or difference file. For example, 321 means Version 3 Release 2 Modification 1.

Comp

Identifies the component to which the record belongs.

Cat

Identifies the category or subcomponent to which the record belongs.

Appl

Identifies the application to which the record belongs (applicable to records of application dependent components).

Key

Displays the remainder of the record key.

Resetting the MODS Cache

Select option **RC** from the MODS : Definition Utility Menu to reset the MODS cache, the list cache, and the report cache.

This clears all the MODS definitions from the MODS cache, the list definitions from the list cache, and the report definitions from the report cache. Select option **RC** after you have used the MODS definition utility, so that any changes you make take effect. If you do not do this, the changes you have made may not take effect until the next time SOLVE management services is started up.

Part IV

CAS Programming Guide

18

CAS Programming Interface ($CACALL)

This chapter describes the Application Programming Interface (API) to the Common Applications Services (CAS) procedure ($CACALL).

About $CACALL

When $CACALL is invoked, an *action* and a *class* must be specified. These determine which CAS function is invoked. For example, to display a menu, you would invoke $CACALL with the action set to **DISPLAY** and the class set to **MENU**. For the syntax of $CACALL, see the section, *The $CACALL API*, on page 18-3.

$CACALL gives you access to the following functions:

BUILD CFPATH
Builds the MODS control file path.

BUILD CRITERIA
Builds and returns the specified criteria.

BUILD FKA
Builds the specified function key area.

BUILD IDTEXT
Builds lines of text containing details of the specified user ID together with the current date and time.

BUILD MESSAGE
Builds a message from the specified parameters.

DISPLAY DATA
Displays text in browse mode.

DISPLAY HELP
Displays the help text applicable to the user's current context.

DISPLAY LIST
Builds and displays a list from the specified list definition.

DISPLAY MENU
Builds and displays a menu from the specified menu definition.

DISPLAY MESSAGE
Displays the specified message.

EDIT DATA
Displays text in edit mode and returns the edited text to the caller.

EXECUTE COMMAND
Executes the specified command.

GET TENTRY
Retrieve one or all entries from a table.

LOAD COMMAND
Loads the command table.

LOAD TABLE
Loads the specified table.

LOAD PDOMAIN
Loads the specified panel domain.

NAVIGATE PDOMAIN
Determines the next panel to be displayed.

VALIDATE DATA
Validates data according to the specified parameters.

The $CACALL API

The syntax for invoking $CACALL is as follows:

```
-EXEC $CACALL     OPT=ACTION
                  ACTION={BUILD | DISPLAY | EDIT | EXECUTE | GET.|
                          LOAD | NAVIGATE | VALIDATE}
                  CLASS={CFPATH | COMMAND | CRITERIA | DATA |
                         FKA | HELP | IDTEXT | LIST | MENU |
                         MESSAGE | PDOMAIN | TABLE | TENTRY}
                  [NAME='attribute1=value1 attribute 2=value2 ...
                        attributen=valuen']
                  [PARMS='parm1=value1 parm2=value2 ...
                         parmn=valuen']
```

Operands

OPT=ACTION
An optional parameter indicating the option required.

ACTION={BUILD | DISPLAY | EDIT | EXECUTE | GET | LOAD | NAVIGATE | VALIDATE}
A required parameter indicating the action to perform.

CLASS={CFPATH | COMMAND | CRITERIA | DATA | FKA | HELP | IDTEXT | LIST | MENU | MESSAGE | PDOMAIN | TABLE | TENTRY}
A required parameter that identifies the class on which the action is to be performed.

NAME='*attribute1=value1 attribute2=value2 ... attributen=valuen*'
An optional parameter that gives the attribute IDs and values which identify the object that is to be processed.

PARMS='*parm1=value1 parm2=value2 ... parmn=valuen*'
Optional keyword parameters. These parameters are documented with the individual calls.

Note

If any value (that is, value1..*n*) specified in the NAME or PARMS operand contains imbedded blanks, quotes, or double quotes, then the value and the entire operand must be quoted as described in the &ZQUOTE verb description in the *Network Control Language Reference* manual.

Input Variables

Any required input variables are described for the individual calls.

Return Variables

Any variables set by $CACALL are described for the individual calls.

Return Codes

$CACALL sets one of two possible return codes:

0	OK
8	Error

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

0	No feedback code set
1	Definition not found
2	No data found
3	Cancelled by exit
4	Cancelled by user
5	End of sequence
6	Unable to obtain lock
7	User not authorized
8	Processing error
9	Return requested
10	Nesting level exceeded
11	Definition not eligible for processing

Action : BUILD
Class : CFPATH

Function

Builds a MODS control file path. The path enables control file concatenation allowing multiple control files to be accessed concurrently. The path accommodates up to five separate control files.

This function is usually invoked only by the SOLVE management services initialization procedure (NMINIT).

```
&CONTROL SHRVARS=($DB)
-EXEC $CACALL      OPT=ACTION
                   ACTION=BUILD
                   CLASS=CFPATH
                   NAME='PATHNAME=pathname'
```

Operands

PATHNAME=*pathname*

A required parameter specifying the name by which the pathname is identified.

Input Variables

Note

You must specify the input variables in sets; each set consists of variables ending in the same numeral (for example, $DBDDNAME1, $DBLIB1, $DBDSN1, $DBDESC1, and $DBOPEN1.

$DBDDNAME1..5

The DD name(s) to which specified datasets are ALLOCATED. At least one DD name must be specified.

$DBLIB1..5

The file ID(s) to be associated with the control files. At least one file ID must be specified.

$DBDSN1..5

The dataset name(s) of the control files. At least one dataset name must be specified.

$DBDESC1..5

Optional description(s) of the file(s).

$DBOPEN1..5

Optional operands to be applied when the files are opened.

Return Variables

&SYSMSG

System Message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8	Processing error
10	Nesting level exceeded

Example

The following statements build a 2-level MODS control file path. The first (top) level file is MODSUSR and is modifiable; the second level file is MODSDIS and is not modifiable.

```
&$DBLIB1 = MODSUSR
&$DBDDNAME1 = MODSUSR
&$DBDSN1 = &&000$NMDSNQ.MODSUSR
&$DBDESC1 = &STR 'User MODS Library'
&$DBOPEN1 = &STR 'LSR EXIT=NO'
&$DBLIB2 = MODSDIS
&$DBDDNAME2 = MODSDIS
&$DBDSN2 = &&000$NMDSNQ.MODSDIS
&$DBDESC2 = &STR 'Distributed MODS Library - Read only'
&$DBOPEN2 = &STR 'LSR INPUT EXIT=NO'
&CONTROL SHRVARS=($DB)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=BUILD +
                 CLASS=CFPATH +
                 NAME='PATHNAME=MODSCTL'
```

Action : BUILD
Class : CRITERIA

Function

Builds a criteria using the criteria definition. You can supply values for the variables used in the definition. Items can be tested against the built criteria.

```
&CONTROL SHRVARS=($CR,pref,...pref)

-EXEC $CACALL      OPT=ACTION
                   ACTION=BUILD
                   CLASS=CRITERIA
                   NAME='APPL=application id
                         [TYPE={PUBLIC | PRIVATE | FREEFORM}]
                         [USER=userid]
                         NAME=criteria name'
                   PARMS='MESSAGE=message'
```

Operands

APPL=*application id*

A required parameter (not applicable for TYPE=FREEFORM) giving the application identifier of the criteria.

TYPE={PRIVATE | PUBLIC}

An optional parameter giving the type of the criteria. Valid values are:

PUBLIC

Public criteria—available for general use.

PRIVATE

Private criteria—owned by a specific user ID.

FREEFORM

Free-form criteria—presents a panel for the user to enter criteria.

Note

If you do not specify TYPE or USER, the function attempts to find a PRIVATE definition owned by the invoking user ID first. If unsuccessful, the function uses a PUBLIC definition.

USER=*userid*

An optional parameter (not applicable for TYPE=PUBLIC and TYPE=FREEFORM) giving the user ID of the user owning the criteria. Default is the user ID of the user invoking the function.

NAME=*criteria name*

A required parameter (not applicable for TYPE=FREEFORM) giving the name of the criteria.

MESSAGE=*message*

A message to be displayed on the criteria panel, if one is defined.

Input Variables

Variables with prefixes as specified in SHRVARS.

Return Variables

&$CRCRIT*nnnn*

The criteria line(s).

&$CRCRITTOTAL

The total number of criteria lines built (up to 9999). The value gives the number of &$CRCRIT*nnnn* variables.

&$CRPANEL

Set to **YES** or **NO** indicating whether a run time panel is defined.

&$CRPANELCMD

Set to **EXIT** or **ACTION** indicating the command executed to terminate the run time panel, if a run time panel is defined, otherwise set to null.

&SYSMSG

System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

1	Definition not found
3	Cancelled by exit
8	Processing error
10	Nesting level exceeded

Example

The following statements build the public criteria ZPRPR001 for the application identified by the ID ZPR:

```
&CONTROL SHRVARS=($CR,ZPR)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=BUILD +
                 CLASS=CRITERIA +
                 NAME=‘  APPL=ZPR +
                         TYPE=PUBLIC +
                         NAME=ZPRPR001’
```

Action : BUILD
Class : FKA

Function

Builds a function key area. You can build the function key area by using a predefined one or from scratch. Your procedure should call the BUILD FKA function before displaying a panel.

```
&CONTROL SHRVARS=($FK)

-EXEC $CACALL      OPT=ACTION
                   ACTION=BUILD
                   CLASS=FKA
                   [NAME='FKA=function key area']
```

Operands

FKA=*function key area*

An optional parameter identifying the predefined function key area on which to base the new function key area.

See *Predefined Function Key Areas*, on page 18-12 for details.

Input Variables

&$FK1...&$FK24

The function key *actions* for any or all of the keys F1 to F24. Specify NOACT to inactivate and remove a function key from the function key area.

&$FKLAB1...&$FKLAB24

The *labels* that are displayed in the function key area (the bottom two lines of the displayed screen). Each label can be up to 8 characters in length. If not specified, the label for a key defaults to the first word of the key's action.

&$FKS1...&$FKS24

These variables are used internally by the function. *Do not* alter the values of these variables.

&$FKSLAB1...&$FKSLAB24

These variables are used internally by the function. *Do not* alter the values of these variables.

&$FKOPTS

This variable is used internally by the function. *Do not* alter the value of this variable.

Return Variables

&$FK1...&$FK24
The function key actions for keys F1 to F24.

&$FKLAB1...&$FKLAB24
The function key labels for keys F1 to F24.

&$FKS1...&$FKS24
These variables are used internally by the function. ***Do not*** alter the values of these variables.

&$FKSLAB1...&$FKSLAB24
These variables are used internally by the function. ***Do not*** alter the values of these variables.

&$FKOPTS
This variable is used internally by the function. ***Do not*** alter the value of this variable.

&$FKA1
Function key area line 1.

&$FKA2
Function key area line 2.

Note

Depending on the user's current function key area display option, **&$FKA1** and **&$FKA2** can contain F1 through F12, F13 through F24, or nothing.

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8 Processing error

10 Nesting level exceeded

Example

The following statements build a function key area by using the predefined function key area $CAADD, explicitly setting F5 to HISTORY and F6 to TEXT, and disabling F7 and F11:

```
&$FK5 = HISTORY
&$FK6 = TEXT
&$FK7 = NOACT
&$FK11 = NOACT
&CONTROL SHRVARS=($FK)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=BUILD +
                 CLASS=FKA +
                 NAME='   FKA=$CAADD'
```

Predefined Function Key Areas

The following table shows the function key settings of the predefined function key areas:

Function Key Areas	Function Key Actions											
	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
$CAADD	HELP	SPLIT	FILE	SAVE			BACKWARD	FORWARD	SWAP			CANCEL
$CABRO	HELP	SPLIT	EXIT				BACKWARD	FORWARD	SWAP			
$CADEL	HELP	SPLIT					BACKWARD	FORWARD	SWAP			CANCEL
$CAUPD	HELP	SPLIT	FILE	SAVE			BACKWARD	FORWARD	SWAP			CANCEL
$IMADD	HELP	SPLIT	FILE			TEXT	BACKWARD	FORWARD	SWAP	CANCEL		CANCEL
$IMBRO	HELP	SPLIT	EXIT		HISTORY	TEXT	BACKWARD	FORWARD	SWAP			
$IMDEL	HELP	SPLIT	CANCEL						SWAP			
$IMUPD	HELP	SPLIT	FILE		HISTORY	TEXT	BACKWARD	FORWARD	SWAP	CANCEL		CANCEL

Action : BUILD
Class : IDTEXT

Function

Builds the ID text that contains information about a user.

```
&CONTROL SHRVARS=($CAID)

-EXEC $CACALL       OPT=ACTION
                    ACTION=BUILD
                    CLASS=IDTEXT
                    [NAME='USER=userid']
                    [PARMS='[BORDER={YES | NO}]
                              [LENGTH={80 | length}]
                              [DATE=date]
                             [TIME=time]']
```

Operands

USER=user*id*

An optional parameter giving the ID of the user whose ID text is to be built. The default is the current user ID.

BORDER={YES | NO}

An optional parameter specifying whether there are borders at the top and bottom of the built ID text. The default is YES.

LENGTH={80 | *length*}

An optional parameter specifying the length of the horizontal border lines if BORDER=YES. The default is 80, and the range is 1 through 255.

DATE=*date*

An optional parameter specifying the date (in DATE3 format) to be included in the built ID text. The default is the current date.

TIME=*time*

An optional parameter specifying the time (in *hh.mm.ss* format) to be included in the built ID text. The default is the current time.

Input Variables

None.

Return Variables

&$CAIDTX*n*
Contains the built ID text strings.

&$CAIDTXTOT
Contains the number of ID text strings built. The value gives the number of &$CAIDTX*n* variables.

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8 Processing error

10 Nesting level exceeded

Example

The following statements build the ID text for the current user. The text contains the current date and time, and has 80-character borders at the top and bottom.

```
&CONTROL SHRVARS=($CAID)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=BUILD +
                 CLASS=IDTEXT +
                 NAME='   USER=&USERID' +
                 PARMS='  BORDER=YES +
                          LENGTH=80'
```

Action : BUILD
Class : MESSAGE

Function

Builds a message using a message definition. The BUILD MESSAGE function returns the built message in &SYSMSG.

```
&CONTROL NOSHRVARS

-EXEC $CACALL      OPT=ACTION
                   ACTION=BUILD
                   CLASS=MESSAGE
                   NAME='MESSAGE=message id'
                   PARMS='PROCNAME=procedure name
                          [P1=parameter1...
                          P10=parameter10]'
```

Operands

MESSAGE=*message id*
A required parameter giving the identifier of the message to be built.

PROCNAME=*procedure name*
A required parameter giving the name of the procedure that requested the message to be built.

P1=*parameter1*...P10=*parameter10*
Optional parameters providing data values to be substituted into the message definition.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains either:

- The requested message (for return code 0)
- The error message (for return code 8)

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

1	Definition not found
8	Processing error
10	Nesting level exceeded

Example

The message definition for message SAMP01 is

```
VSAM ERROR ON FILE ~P1 RC=~P2 FDBK=~P3
```

Use the following statements to build the SAMP01 message:

```
&CONTROL NOSHRVARS
-EXEC $CACALL    OPT=ACTION
                 ACTION=BUILD
                 CLASS=MESSAGE
                 NAME='   MESSAGE=SAMP01'
                 PARMS='  PROCNAME=&0
                          P1=&FILEID
                          P2=&FILERC
                          P3=&VSAMFDBK'
```

&0 contains the name of your procedure; &FILEID, &FILERC, and &VSAMFDBK contain, respectively, the values for `~P1`, `~P2`, and `~P3` in the message definition.

Action : DISPLAY
Class : DATA

Function

Displays text in browse mode. The DISPLAY DATA function provides text browsing facilities for up to 9999 lines of text. When a user requests help, help text associated with the specified application ID and function name is displayed.

```
&CONTROL SHRVARS=($ED)

-EXEC $CACAL        OPT=ACTION
                    ACTION=DISPLAY
                    CLASS=DATA
                    PARMS='APPL=application id
                              FUNC=function name
                              [TITLE=title]
                              [LINECNT={0 | n}]
                              [LINELEN={256 | n}]
                              [LINENUM={YES | NO}]
                              [INDENT={0 | n}]
                              [TOPEXIT={YES | NO}]
                              [MESSAGE=message]
                              [USERFUNC=function]
                              [LINETOP={1 | n}]
                              [CANCEL={YES | NO}]
                              [LMARGIN={1 | n}]
                             [RMARGIN=n]'
```

Operands

APPL=*application id*

A required parameter giving the application identifier.

FUNC=*function name*

A required parameter indicating the function being performed.

TITLE=*title*

An optional parameter giving the title to be displayed at the top of the panel. The default is CAS : Text Editor.

LINECNT={0 | *n*}

A required parameter giving the total number of lines of text. The default is 0, and the range is 0 through 9999.

LINELEN={256 | *n*}

An optional parameter indicating the maximum line length. The default is 256, and the range is 1 through 256.

LINENUM={YES | NO}

An optional parameter that indicates whether to display line numbers. The default is YES.

INDENT={0 | *n*}

An optional parameter that indicates the number of positions to indent the text lines. The default is 0, and the range is 0 through 256 minus the value in LINELEN.

TOPEXIT={YES | NO}

An optional parameter indicating whether executing the BACKWARD command—when the display is positioned at the top of the text—ends the text display and causes control to be returned to the caller. The default is NO.

MESSAGE=*message*

An optional parameter specifying a message to be displayed on line 3 of the panel, on initial entry.

USERFUNC=*function*

An optional parameter specifying the logical function being performed (for example, Browse). If specified, the function is displayed on line 4 of the panel as Function=*function*. *function* must not be longer than eight characters.

LINETOP={1 | *n}*

An optional parameter specifying the number of the line to be displayed as the first line of text, on initial entry. The default is 1, and the range is 1 to the value in LINECNT.

CANCEL={YES | NO}

An optional parameter specifying whether the CANCEL command is supported. The default is NO.

LMARGIN={1 | *n*}

The left hand margin (in characters) used by the text editor. The default is 1, and the range is 1 through the value in LINELEN.

RMARGIN=*n*

The right hand margin (in characters) used by the text editor. The default is the lesser of the value in LINELEN and the logical screen width. The range is the value of LMARGIN plus 20 to the value in LINELEN.

Input Variables

&$EDFK1...&$EDFK24

The function key *actions* for any or all of the keys F1 to F24. Specify NOACT to inactivate and remove a function key from the function key area.

&$EDFKLAB1...&$EDFKLAB24

The *labels* that are displayed in the function key area (the bottom two lines of the displayed screen). Each label can be up to 8 characters in length. If not specified, the label for a key defaults to the first word of the key's action.

&$EDCOMMENT*n*

Contains the comment lines to be displayed (up to 9) above the text lines.

&$EDLINE*nnnn*

Contains the text lines to be displayed (up to 9999).

&$EDEXITCMDS=*command1, command2, ...*

Specifies commands which are accepted as valid exit commands. That is, if a specified command is executed, control is returned to the caller.

Return Variables

&$EDCMDEXIT

The command which was entered by the user to exit (normally EXIT).

&$EDCMDPARMS

The parameters for the exit command in &$EDCMDEXIT.

&$EDCOMMAND

The entire contents of the exit command including parameters.

&$EDLINETOP

The line number at the top of the display on exit.

&SYSMSG

System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8	Processing error
10	Nesting level exceeded

Example

The following statements display the contents of &$EDLINE1 and &$EDLINE2 in two lines on a panel titled Problem Text:

```
&$EDLINE1 = &STR Problem text line 1
&$EDLINE2 = &STR Problem text line 2
&CONTROL SHRVARS=($ED)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=DISPLAY +
                 CLASS=DATA +
                 PARMS='  APPL=ZPR +
                          FUNC=BROWSE +
                          TITLE="Problem Text" +
                          LINECNT=2'
```

Action : DISPLAY
Class : HELP

Function

Displays online help. You determine the application, the function, the field, and the window on which the user requires help, and supply the values to the DISPLAY HELP function to display the appropriate help. If the required help has not been added, the function displays the next more general help in the order field-level help, window-level help, function-level help, and application-level help.

```
&CONTROL NOSHRVARS

-EXEC $CACALL       OPT=ACTION
                    ACTION=DISPLAY
                    CLASS=HELP
                    NAME='APPL=application id
                          [FUNC=function name]
                          [FIELD=field name]'
                    [PARMS='CROW=cursor row
                            CCOL=cursor column'
                           MODE={VIEW | BROWSE}]
```

Operands

APPL=*application id*

A required parameter giving the identifier of the application for which help is to be displayed.

FUNC=*function name*

An optional parameter (required if either the **FIELD=**, **CROW=** or **CCOL=** keywords are specified) giving the name of the function for which help is to be displayed.

FIELD=*field name*

An optional parameter giving the name of the field for which help is to be displayed.

CROW=*cursor row*

An optional parameter giving the cursor row position when help was requested. This defaults to the value of &CURSROW.

CCOL=*cursor column*

An optional parameter giving the cursor column position when help was requested. This defaults to the value of &CURSCOL.

MODE={VIEW | BROWSE}

If browse is specified, then the help is displayed in *Browse* mode.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8 Processing error

10 Nesting level exceeded

Example

The following statements display the field-level help for the ZPRSTATUS field for the UPDATE function in the application identified by the identifier ZPR if the help is available; otherwise, the next more general help is displayed:

```
&CONTROL NOSHRVARS
-EXEC $CACALL    OPT=ACTION +
                 ACTION=DISPLAY +
                 CLASS=HELP +
                 NAME=‘  APPL=ZPR +
                         FUNC=UPDATE +
                         FIELD=ZPRSTATUS’
```

Action : DISPLAY
Class : LIST

Function

Displays a list.

```
&CONTROL SHRVARS=($LH)

-EXEC $CACALL      OPT=ACTION
                   ACTION=DISPLAY
                   CLASS=LIST
                   NAME='APPL=application id
                         [TYPE={PUBLIC | PRIVATE}]
                         [USER=userid]
                         NAME=list name'
                   [PARMS='[FORMAT={ACTION | MSELECT |
                                    SSELECT | NUMBERED}]
                     [CRITERIA=criteria]
                     [MAXSEL={9999 | nnnn}]
                    [AUTOSEL={YES | NO}]']
```

Operands

APPL=*application id*

A required parameter giving the application identifier of the list.

TYPE={PRIVATE | PUBLIC}

An optional parameter giving the type of list. Valid values are:

PUBLIC

Public list—available for general use.

PRIVATE

Private list—owned by a specific user ID.

Note

If you do not specify TYPE or USER, the function attempts to find a PRIVATE list owned by the invoking user ID first. If unsuccessful, the function uses a PUBLIC list.

USER=*userid*

An optional parameter (if TYPE is not PUBLIC) giving the user ID of the user owning the list. Default is the user ID of the user invoking the function.

NAME=*list name*

A required parameter giving the name of the list.

FORMAT={<u>ACTION</u> | MSELECT | SSELECT| NUMBERED}

An optional parameter indicating the format in which the list is to be displayed. Valid values are:

ACTION
Action list

MSELECT
Multiple selection list

SSELECT
Single selection list

NUMBERED
Numbered list

The default is ACTION.

CRITERIA=*criteria*

A criteria statement used to select items to go in the list. The format must conform to that required by the service procedure specified in the list definition. If specified, this criteria overrides any criteria specified in the list definition, and no &$LHCRIT*nnnn* variables can be set.

MAXSEL={<u>9999</u> | *nnnn*}

The maximum number of items that can be chosen from a list in format MSELECT (multiple selection list). The default is 9999, and the range is 1 through 9999.

AUTOSEL={YES | <u>NO</u>}

Determines whether an entry in a list is automatically selected if it is the only entry in the list. The default is NO.

Input Variables

&$LHCRIT*nnnn*

Criteria variable(s) (up to 9999 variables can be given). Cannot be specified if the **CRITERIA** operand is specified. The format must conform to that required by the service procedure specified in the list definition.

Return Variables

&$LHENTTOTAL

The total number of entries selected (up to 9999).

&$LHENTID*nnnn*

The ID(s) of the list entries selected.

&SYSMSG

System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

1	Definition not found
2	No data found
3	Cancelled by exit
4	Cancelled by user
8	Processing error
9	Return requested
10	Nesting level exceeded
11	Definition not eligible for processing

Example

The following statements display action list ZPRPRALL in the application identified by the ID ZPR. A private list is displayed in preference to a public list.

```
&CONTROL SHRVARS=($LH)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=DISPLAY +
                 CLASS=LIST +
                 NAME='   APPL=ZPR +
                 NAME=ZPRPRALL' +
                 PARMS='  FORMAT=ACTION'
```

Action : DISPLAY
Class : MENU

Function

Displays a menu.

```
&CONTROL NOSHRVARS

-EXEC $CACALL        OPT=ACTION
                     ACTION=DISPLAY
                     CLASS=MENU
                     NAME='APPL=application id
                           MENU=menu id'
                     [PARMS='PSKIP=xxx.xxx']
```

Operands

APPL=*application id*
A required parameter giving the application identifier.

MENU=*menu id*
A required parameter giving the menu identifier.

PSKIP=*xxx.xxx*
Panel skip setting.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8	Processing error
9	Return requested
10	Nesting level exceeded

Example

The following statements display menu 030 in the application identified by the ID ZPR:

```
&CONTROL NOSHRVARS
-EXEC $CACALL    OPT=ACTION +
                 ACTION=DISPLAY +
                 CLASS=MENU +
                 NAME=‘   APPL=ZPR +
                          MENU=030’
```

Action : DISPLAY
Class : MESSAGE

Function

Displays a message and its associated explanation, system action and user action.

```
&CONTROL NOSHRVARS
-EXEC $CACALL      OPT=ACTION
                   ACTION=DISPLAY
                   CLASS=MESSAGE
                   NAME='MESSAGE=message id'
```

Operands

MESSAGE=*message id*
A required parameter giving the identifier of the message to be displayed.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

1	Definition not found
8	Processing error
10	Nesting level exceeded

Example

The following statements display information on message PV7014:

```
&CONTROL NOSHRVARS
-EXEC $CACALL    OPT=ACTION +
                 ACTION=DISPLAY +
                 CLASS=MESSAGE +
                 NAME='   MESSAGE=PV7014'
```

Action : EDIT
Class : DATA

Function

Displays text in edit mode. The EDIT DATA function provides editing facilities for up to 9999 lines of text, each of up to 256 characters in length. When a user requests help, help text associated with the specified application ID and function name is displayed.

```
&CONTROL SHRVARS=($ED)

-EXEC $CACALL        OPT=ACTION
                     ACTION=EDIT
                     CLASS=DATA
                     PARMS='APPL=application id
                              FUNC=function name
                              [TITLE=title]
                              [LINECNT={0 | n}]
                              [LINELEN={256 | n}]
                              [MAXLINES={9999 | n}]
                              [EDTLINES=n]
                              [TOPEXIT={YES | NO}]
                              [MESSAGE=message]
                              [USERFUNC=function]
                              [LINETOP={1 | n}]
                              [LMARGIN={1 | n}]
                             [RMARGIN=n]'
```

Operands

APPL=*application id*

A required parameter giving the application identifier.

FUNC=*function name*

A required parameter indicating the function being performed.

TITLE=*title*

An optional parameter giving the title to be displayed at the top of the panel. The default is CAS : Text Editor.

LINECNT={0 | *n*}

A required parameter giving the number of lines of text. The default is 0, and the range is 0 through 9999.

LINELEN={256 | *n*}

An optional parameter indicating the maximum line length. The default is 256, and the range is 1 through 256.

MAXLINES={9999 | *n*}

An optional parameter indicating the maximum number of lines of text that can exist. The default is 9999, and the range is 1 through 9999.

EDTLINES=*n*

An optional parameter giving the number of lines that can be edited. If specified, the first *n* lines can be edited. If not specified, the number of lines that can be edited defaults to the value of the LINECNT parameter—that is, all lines can be edited. The range is 0 to the value in LINECNT.

TOPEXIT={YES | NO}

An optional parameter indicating whether executing the BACKWARD command—when the display is positioned at the top of the text—ends the text display and causes control to be returned to the caller. The default is NO.

MESSAGE=*message*

An optional parameter specifying a message to be displayed on line 3 of the panel, on initial entry.

USERFUNC=*function*

An optional parameter specifying the logical function being performed (for example, Update). If specified, the function is displayed on line 4 of the panel as Function=*function*. *function* must not be longer than eight characters.

LINETOP={1 | *n*}

An optional parameter specifying the number of the line to be displayed as the first line of text, on initial entry. The default is 1, and the range is 1 through the value in LINECNT.

LMARGIN={1 | *n*}

The left hand margin (in characters) used by the text editor. The default is 1, and the range is 1 through the value in LINELEN.

RMARGIN=*n*

The right hand margin (in characters) used by the text editor. The default is the lesser of the value in LINELEN and the logical screen width. The range is the value of LMARGIN plus 20 to the value in LINELEN.

Input Variables

&$EDFK1...&$EDFK24

The function key *actions* for any or all of the keys F1 to F24. Specify NOACT to inactivate and remove a function key from the function key area.

&$EDFKLAB1...&$EDFKLAB24

The *labels* that are displayed in the function key area (the bottom two lines of the displayed screen). Each label can be up to 8 characters in length. If not specified, the label for a key defaults to the first word of the key's action.

&$EDCOMMENT*n*

Contains the comment lines to be displayed (up to 9) above the text lines.

&$EDLINE*nnnn*

Contains the text lines to be displayed for edit (up to 9999).

&$EDEXITCMDS=*command1, command2, ...*

Specifies commands which are accepted as valid exit commands. That is, if a specified command is executed, control is returned to the caller.

Return Variables

&$EDLINE*nnnn*

The edited text lines.

&$EDLINECNT

The number of edited text lines. The value gives the number of &$EDLINE*nnnn* variables.

&$EDMODIFIED

Whether any lines were modified (YES or NO).

&$EDCMDEXIT

The command which was entered by the user to exit (normally FILE or SAVE).

&$EDCMDPARMS

The parameters for the exit command in &$EDCMDEXIT.

&$EDCOMMAND

The entire contents of the exit command including parameters.

&$EDLINETOP

The line number at the top of the display on exit.

&SYSMSG

System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

4 Cancelled by user

8 Processing error

10 Nesting level exceeded

Example

The following statements display the contents of &$EDLINE1 and &$EDLINE2 in two lines on a panel titled Problem Text for editing:

```
&$EDLINE1 = &STR Problem text line 1
&$EDLINE2 = &STR Problem text line 2
&CONTROL SHRVARS=($ED)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=EDIT +
                 CLASS=DATA +
                 PARMS='  APPL=ZPR +
                          FUNC=UPDATE +
                          TITLE="Problem Text" +
                          LINECNT=2'
```

Action : EXECUTE
Class : COMMAND

Function

Executes a command if it is known. When a user requests help, help text associated with the specified application ID, function name, and message ID (if specified) is displayed.

```
&CONTROL SHRVARS=($FK,$CM)

-EXEC $CACALL       OPT=ACTION
                    ACTION=EXECUTE
                    CLASS=COMMAND
                    [NAME='COMMAND=command']
                    [PARMS='APPL=application id]
                              [FUNC=function name]
                              [KEY=key]
                              [MESSAGE=messageid]
                             [CURSFLD=cursorfield value]']
```

Operands

APPL=*application id*

An optional parameter giving the application identifier (required for the HELP command).

FUNC=*function name*

An optional parameter indicating the function currently being performed (required for the HELP command).

COMMAND=*command*

An optional parameter giving the command to be executed (required if the KEY keyword is not used). If you use both the COMMAND and KEY keywords, and a command is assigned to the function key specified in KEY, the command assigned to the function key takes precedence. The following commands (or any command defined using the CAS Command Definition facility) can be specified:

CMD	Invoke the SOLVE Command Entry facility
EX[EC]	Execute an NCL procedure
DISC[ONN]	Disconnect the session
HELP	Display help information
KEYS	Switch FKA display
LOCK	Lock the session
NOTEPAD	Display the CAS notepad
PASSWORD	Set user password
PQ[UEUE]	Display PSM print queue

PSKIP	Skip panels
RETRIEVE	Retrieve the last command
SPLIT	Split window
START	Start an NCL procedure
SWAP	Swap window
WHERE	Display NCL procedure details

SOLVE commands are described in full in the *SOLVE Command Reference*.

KEY=*key*

An optional parameter giving the last function key (&INKEY value) that was pressed (required if the COMMAND keyword is not used). If you use both the COMMAND and KEY keywords, and a command is assigned to the function key specified in KEY, the command assigned to the function key takes precedence.

MESSAGE=*message id*

An optional parameter giving the message identifier of the currently displayed message (for example, when a user requires help on a message).

CURSFLD=*cursorfield value*

An optional parameter containing the value of the field in which the cursor is located.

Input Variables

None.

Return Variables

&$FK1...&$FK24

The function key actions for keys F1 to F24.

&$FKLAB1...&$FKLAB24

The function key labels for keys F1 to F24.

&$FKA1

Function key area line 1.

&$FKA2

Function key area line 2.

&$CMDI

Set to **Y** or **N** to indicate if an action was performed. For example, a command is not executed if it is not in the command table.

&$CMDS
Set to **C**, **K**, or **N** indicating the source of the action (Command, Key, or None).

&$CMD
The first word of the action.

&$CMDPARMS
The remaining operands or parameters of the action.

&$CMDR
The retrieved command (when COMMAND=RETRIEVE).

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

7	User not authorized
8	Processing error
9	Return requested
10	Nesting level exceeded

Example

The following statements execute the command in variable **&COMMAND** or assigned to the key indicated in variable **&INKEY**; the APPL, FUNC, and MESSAGE keywords are specified and are referenced if the command is HELP.

```
&CONTROL SHRVARS=($FK,$CM)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=EXECUTE +
                 CLASS=COMMAND +
                 NAME=‘   COMMAND=&COMMAND’ +
                 PARMS=‘  APPL=ZPR +
                          FUNC=UPDATE +
                          KEY=&INKEY +
                          MESSAGE=PV7014’
```

Action : LOAD
Class : COMMAND

Function

Loads the command table into memory (normally during system initialization). You must perform the LOAD COMMAND function before you can execute a defined command.

```
&CONTROL NOSHRVARS
-EXEC $CACALL       OPT=ACTION
                    ACTION=LOAD
                    CLASS=COMMAND
```

Operands

None.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

2	No data found
6	Unable to obtain lock
8	Processing error
10	Nesting level exceeded

Example

The following statements load the command table:

```
&CONTROL NOSHRVARS
-EXEC $CACALL    OPT=ACTION +
                 ACTION=LOAD +
                 CLASS=COMMAND
```

Action : GET
CLASS : TENTRY

Function

Retrieves one or all entries from a table.

```
&CONTROL SHRVARS=($VM)

-EXEC $CACALL       OPT=ACTION
                    ACTION=GET
                    CLASS=TENTRY
                    NAME='APPL=application id
                          FIELD=fieldName
                          [VALUE=fullValue]'
                    [PARMS='ACTIVE={YES | NO | ANY}']
```

Operands

APPL=*application id*
A required parameter giving the identifier of the application.

FIELD=*fieldName*
A required parameter giving the name of the table.

VALUE=*fullValue*
An optional parameter giving the full value of the entry to be retrieved. If omitted, all entries in the table are returned.

ACTIVE={YES | NO | ANY}
An optional parameter that indicates which entries are to be retrieved from the table if the VALUE parameter is omitted.

YES	Only active table entries are to be retrieved. This is the default.
NO	Only inactive table entries are to be retrieved.
ANY	All table entries are to be retrieved.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains the error message (for return code 8).

If a full value is supplied, the following variables are returned:

$VMABBR
The abbreviated value of the table entry.

$VMFULL
The full value of the table entry.

$VMDESC
The description of the table entry.

$VMTEXT*n*
The text fields of the table entry.

If a full value is not supplied, the following variables are returned:

$VMTOTAL
The total number of entries in the table.

$VMABBR*n*
The abbreviated value for each table entry.

$VMFULL*n*
The full value for each table entry.

$VMDESC*n*
The description of each table entry.

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

2	No data found
8	Processing error
10	Nesting level exceeded

Example

The following statements retrieve the entry $NDSYS from the Report Writer table:

```
&CONTROL SHRVARS
-EXEC $CACALL    OPT=ACTION +
                 ACTION=GET +
                 CLASS=TENTRY +
                 NAME='   APPL=$RW +
                          FIELD=APPL +
                          VALUE=$NDSYS'
```

Action : LOAD
Class : PDOMAIN

Function

Loads panel domains (normally during the initialization of an application). You must perform the LOAD PDOMAIN function before you can use the NAVIGATE PDOMAIN function.

```
&CONTROL NOSHRVARS

-EXEC $CACALL       OPT=ACTION
                    ACTION=LOAD
                    CLASS=PDOMAIN
                    NAME='APPL=application id
                          [TYPE={PUBLIC | PRIVATE}]
                          [USER=userid]
                         [NAME=domain name]'
```

Operands

APPL=*application id*
A required parameter giving the identifier of the application.

TYPE={PUBLIC | PRIVATE}
An optional parameter giving the type of panel domain. Valid values are:

PUBLIC
Public domain—available for general use.

PRIVATE
Private domain—owned by a specific user ID.

USER=*userid*
An optional parameter (if TYPE is not PUBLIC) giving the user ID of the user owning the panel domains.

NAME=*domain name*
An optional parameter giving the name of the panel domain.

Note
The LOAD PDOMAIN function loads all panel domains matching the supplied operands.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8 Processing error

10 Nesting level exceeded

Example

The following statements load the panel domains for the application identified by ID ZPR:

```
&CONTROL NOSHRVARS
-EXEC $CACALL    OPT=ACTION +
                 ACTION=LOAD +
                 CLASS=PDOMAIN +
                 NAME='   APPL=ZPR'
```

Action : LOAD
Class : TABLE

Function

Loads the tables for an application (normally during system initialization). You must perform the LOAD TABLE function before you can the tables to validate data.

```
&CONTROL NOSHRVARS

-EXEC $CACALL       OPT=ACTION
                    ACTION=LOAD
                    CLASS=TABLE
                    NAME='APPL=application id
                          [FIELD=fieldName]'
                    [PARMS='RELOAD={YES | NO}']
```

Operands

APPL=*application id*
A required parameter giving the identifier of the application.

FIELD=*fieldName*
An optional parameter giving the name of the table if only one table is to be loaded. If omitted, all tables for the specified application are loaded.

RELOAD={YES | NO}
An optional parameter specifying whether the tables for the application are to be reloaded if previously loaded. The default is NO.

Input Variables

None.

Return Variables

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8 Processing error

10 Nesting level exceeded

Example

The following statements load the tables for the application identified by ID ZPR. Any tables already loaded are reloaded.

```
&CONTROL NOSHRVARS
-EXEC $CACALL    OPT=ACTION +
                 ACTION=LOAD +
                 CLASS=TABLE +
                 NAME=‘   APPL=ZPR’ +
                 PARMS=‘  RELOAD=YES’
```

Action : NAVIGATE
Class : PDOMAIN

Function

Determines the next panel to be displayed. You determine the panel by specifying the direction of movement (backward or forward) and the number of panels to skip, or by specifying the name of the panel domain element.

```
&CONTROL SHRVARS=($PV,pref,...pref)

-EXEC $CACALL       OPT=ACTION
                    ACTION=NAVIGATE
                    CLASS=PDOMAIN
                    NAME='APPL=application id
                          [TYPE={PUBLIC | PRIVATE}]
                          [USER=userid]
                          NAME=domain name'
                    [PARMS='[CURRENT=element name]
                             [NEXT={element name | *}]
                             [SKIP={1 | n}]
                            [LOAD={YES | NO}]']
```

Operands

APPL=*application id*
A required parameter giving the application identifier of the panel domain.

TYPE={PRIVATE | PUBLIC}
An optional parameter giving the type of panel domain. Valid values are:

PUBLIC
Public domain—available for general use.

PRIVATE
Private domain—owned by a specific user ID.

Note

If you do not specify TYPE or USER, the function attempts to find the panel domain definition owned by the invoking user ID. If unsuccessful, the function uses a public panel domain definition.

USER=*userid*
An optional parameter (if TYPE=PRIVATE) giving the user ID of the user owning the PRIVATE panel domain. Default is the user ID of the user invoking the function.

NAME=*domain name*
A required parameter giving the name of the panel domain.

CURRENT=*element name*

An optional parameter indicating the current element. If this keyword is blank, the current element are assumed to be ##TOP## (if DIR=FORWARD) or ##END## (if DIR=BACKWARD).

Note

DIR or SKIP is mutually exclusive to NEXT. If you do not specify DIR, NEXT, and SKIP, the default is DIR=FORWARD and SKIP=1.

DIR={FORWARD | BACKWARD}

An optional parameter indicating the direction in which navigation is to occur. The default is FORWARD.

NEXT={*element name* | *}

An optional parameter indicating the next element to progress to. If blank, the next element is determined based upon the defined paths (from the current element) and the direction of travel. If an element name is specified, navigation occurs to this element if it is currently eligible. If '*****' is specified, a pick list of all eligible elements in the domain (excluding the element specified in the CURRENT= keyword) is presented.

SKIP={1 | *n*}

An optional parameter indicating the number of times to perform a panel navigation. The default is 1, and the range is 1 through 9999. Use this parameter when the user specifies a numeric or MAX scroll amount.

LOAD={YES | NO}

An optional parameter determining whether to load a domain if it is not currently loaded. The default is YES.

Input Variables

Variables with prefixes as specified in SHRVARS.

&$PVNEST*nnn*

Tracking variables used by CAS to maintain the last *nnn* elements visited in the current direction (up to 999 elements). These variables are supplied by CAS. Do not modify these variables.

&$PVCRIT*nnnn*

Criteria variables containing the criteria used to specify eligible panel domain element types.

&$PVCRITTOTAL

The total number of criteria variables. The range is 0 through 9999.

Return Variables

&$PVELEMENT
The name of the next element to progress to.

&$PVPANEL
The name of the panel associated with the next element.

&$PVDESC
The description of the next element.

&$PVHELP
The name of the function-level help associated with the next element.

&$PVTYPE
The type (PANEL or TEXT) of the next element.

&$PVTITLE
The title of the next element (if type is TEXT).

&SYSMSG
System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

1	Definition not found
4	Cancelled by user
5	End of sequence
6	Unable to obtain lock
8	Processing error
10	Nesting level exceeded

Example

The following statements identify the next panel for display following the panel in &$PVELEMENT. The panel domain is ZPRPROB in the application identified by ID ZPR. A private panel domain is used in preference to a public panel domain.

```
&CONTROL SHRVARS=($PV,ZPR)
-EXEC $CACALL    OPT=ACTION +
                 ACTION=NAVIGATE +
                 CLASS=PDOMAIN +
                 NAME='   APPL=ZPR +
                 NAME=ZPRPROB' +
                 PARMS='  CURRENT=&$PVELEMENT'
```

Action : VALIDATE
Class : DATA

Function

Validates data against predefined values or edit rules.

```
&CONTROL SHRVARS=(field name,$VM)

-EXEC $CACALL      OPT=ACTION
                   ACTION=VALIDATE
                   CLASS=DATA
                   PARMS='[APPL=application id]
                          FIELD=field name
                          [LIST=n]
                          [TEXT={YES | NO}]
                          [ACTIVE={YES | NO | ANY}]
                          DESC=description
                          EDITS=edit number 1,edit number 2,...
                          [IMSYS=system name]
                         [PROMPT={YES | NO}]'
```

Operands

APPL=*application id*

An optional (required only for EDITS=17) parameter giving the application identifier.

FIELD=*field name*

A required parameter giving the name of the field that is to be validated. This field name must also be included in the **SHRVAR=** operand.

LIST=*n*

An optional parameter (required only for EDITS=18) that gives the number of valid values supplied using the &$VMFULL*nnn* variables. The &$VMABBR*nnn* and &$VMDESC*nnn* variables can also be defined. The range is 1 through 999.

TEXT={YES | NO}

An optional parameter (applicable only for EDITS=17) that indicates whether the text fields associated with the selected or supplied field value are to be returned. The default is NO.

ACTIVE={YES | NO | ANY}

An optional parameter (applicable only for EDITS=17) that indicates which entries from the table defined for the field in FIELD are to be considered as valid values. Valid values are:

YES	Only active table entries are to be considered as valid values. This is the default.

NO	Only inactive table entries are to be considered as valid values.
ANY	All table entries are to be considered as valid values.

DESC=*description*

A required parameter (optional for EDITS=17) that gives the description of the field for use in selection list headings, help panels and error messages.

EDITS=*edit number1,edit number 2,...*

A required parameter giving the edit number(s) that indicate how the field is to be validated (multiple numbers can be specified, separated by commas). Valid values are :

1(*l*) YES/NO

The field can only contain **YES** (or a string beginning with **Y**) or **NO** (or a string beginning with **N**).

If the length parameter ***(l)*** is coded, the field is set to that length. For example, 1(1) returns **Y** or **N** only.

2 Unsigned integer

The field can only contain a positive number with no sign or decimal point.

3 Date

The field can only contain the date format *dd-mmm-yyyy* or a date format described in Appendix D, *Shorthand Time and Date Formats*.

4 Time

The field can only contain the time format *hh.mm* or a time format described in Appendix D, *Shorthand Time and Date Formats*.

SOLVE products now tend to use edit 24 instead of this one.

5 Hexadecimal

The field can only contain values in expanded hexadecimal.

6 Signed numeric

The field can contain a signed or unsigned number with no decimal point.

7 Real number

The field can only contain real numbers. This includes (signed and unsigned) integers, numbers containing a decimal point, and numbers expressed in scientific notation within the range -1E-70 to +1E+70.

8 Name

The field can only contain numbers, alphabetic characters (upper case), and the characters @, # and $. The first character in the field cannot be a number.

9(*l*:*h*) Range

The field can only contain a value within the defined range, where *l* is the lowest valid value and *h* is the highest. If *l* is omitted, the value is only checked for being less than or equal to *h*; if *h* is omitted, the value is only checked for being greater than or equal to *l*.

For example,

- 9(1:10) accepts values from 1 to 10
- 9(:3) accepts values less than or equal to 3
- 9(12:) accepts values greater than or equal to 12

10 Alphanumeric

The field can only contain numbers and alphabetic characters. Lower case characters are converted to upper case.

11 Alphabetic

The field can only contain alphabetic characters. Lower case characters are converted to upper case.

12 National

The field can only contain numbers, alphabetic characters, and the characters @, # and $. Lower case characters are converted to upper case.

13 Dataset name

The field can only contain a valid dataset name, with no quotes. Lower case characters are converted to upper case.

14 No imbedded blanks

The field cannot contain any imbedded blanks.

15(*l*:*h*) Field length

The length of the field must be within the defined range, where *l* is the lowest possible length and *h* is the highest. If *l* is omitted, the length is only checked for being less than or equal to *h*; if *h* is omitted, the length is only checked for being greater than or equal to *l*. The default value for *l* is 1.

16 Scroll amount

The field value must be one of the following valid scroll amounts:

- **MAX** (or a string beginning with M)
- **CSR** (or a string beginning with C)
- **DATA** (or a string beginning with D)

- **PAGE** (or a string beginning with P)
- **HALF** (or a string beginning with H)
- A number between 1 and 99999 (if the string starts with a number, it is truncated at the first non-numeric character)

17 Table
The field is validated against the values held in the table defined for the field in FIELD.

18 List
The field is validated against the values given in &$VMFULL*nnn*, &$VMABBR*nnn*, and &$VMDESC*nnn*.

19 NCL keyword
The field must not contain an NCL keyword.

20 Time
The field can only contain the time format *hh*.*mm*.*ss* or a time format described in Appendix D, *Shorthand Time and Date Formats*.

SOLVE products now tend to use edit 23 instead of this one.

21 Hexadecimal Characters
The field can only contain hexadecimal characters, that is, 0–9 and A–F.

22 IP address
The address must be a valid IP address, in the format *A*.*B*.*C*.*D*, where each of A, B, C, and D have a valid range of 0 to 255.

23 Time
The field must contain the time format *hh*:*mm*:*ss* or a time format described in Appendix D, *Shorthand Time and Date Formats*. The returned string is in the format *hh*:*mm*:*ss*.

SOLVE products now tend to use this edit instead of edit 20.

24 Time
The field can only contain the time format *hh*:*mm* or a time format described in Appendix D, *Shorthand Time and Date Formats*. The returned string is in the format *hh*:*mm*.

SOLVE products now tend to use this edit instead of edit 4.

IMSYS=*system name*
An optional parameter (required for EDITS=17 where the table type is IMFLD or IMREC) that gives the INFO/MASTER system name.

PROMPT={<u>YES</u> | NO}

An optional parameter (applicable only for EDITS=17 or 18) which determines whether a list of valid value is to be displayed if the field value contains a question mark character (?). Specifying NO allows values to be defined which contain a question mark (?), and also allows validation to occur from a non-fullscreen environment.

Input Variables

field name

The data to be validated

&$VMFULL*nnn*

The *full value*(s) of the value(s) (up to 20 characters) against which the field is to be validated (applicable only for EDITS=18). Up to 999 *full values* can be specified.

&$VMABBR*nnn*

The *abbreviation*(s) of the value(s) (up to 8 characters) against which the field is to be validated (applicable only for EDITS=18). Up to 999 *abbreviations* can be specified.

&$VMDESC*nnn*

The *description*(s) of the value(s) (up to 38 characters) against which the field is to be validated (applicable only for EDITS=18). Up to 999 *descriptions* can be specified.

Return Variables

field name

The validated data.

&$VMSELABBR

The abbreviated value of the selected value (applicable only for EDITS=17 or 18).

&$VMSELDESC

The description of the selected value (applicable only for EDITS=17 or 18).

&$VMTEXT1..10

The text associated with the value (applicable only for EDITS=17 and returned only if TEXT=YES).

&SYSMSG

System message. Contains the error message (for return code 8).

Feedback Codes

If a return code of 8 is set, then additional information is available as a feedback code set in &$CAFDBK:

8	Processing error
10	Nesting level exceeded

Example

The following statements validate the &PCODE field against edit numbers 2 (unsigned integer) and 9 (range). The value of &PCODE must be an unsigned integer in the range 1 through 10. In the example, because &PCODE has a value of 12, an error message is returned in &SYSMSG.

```
&PCODE = 12
&DESC = &STR Priority Code
&CONTROL SHRVARS=(PCODE,$VM)
-EXEC $CACALL    OPT=ACTION
                 ACTION=VALIDATE
                 CLASS=DATA
                 PARMS='  APPL=$ML
                          FIELD=PCODE
                          DESC=&DESC
                          EDITS=2,9(1:10)'
```

19

Menu Service Procedure Interface

This chapter describes the menu service procedure interface that is available for Common Application Services (CAS) menus.

About Menu Service Procedure Interface

Menu service procedures are installation written NCL procedures that are available for performing specialized processing on menus.

Menu service procedures provide a convenient means of extending the functionality of menus.

Menu service procedures are optional. You specify the menu service procedure when you define a menu.

A menu service procedure performs installation specific menu processing at defined points during invocation and processing of a menu:

- On initial entry to the menu
- After a user makes a selection but before actioning it
- After a user's selection is actioned
- Prior to exiting the menu
- When a user enters an unrecognized command
- If the menu times out when an INWAIT value is specified

You write your own menu service procedures by using the variables described in the following sections.

The variables can be divided into two groups as follows:

- The first group of variables provide information to the procedure and are *not* modifiable (read-only). In particular, you should check the value of &$MHOPT to determine the stage of menu processing for you to implement the required special processing.
- The second group of variables are modifiable and enable the procedure to return information to the system.

You can use your own variables in the menu service procedure. However, do not use variable names that start with #MH. The menu service procedure is called with NOSHRVARS=(#MH). Once you specify your own variables, these variables are persistent through the different stages of menu processing. That is, if you specify &A = ***variable-value***, the value of this variable is available on each subsequent call to the procedure.

$MHOPT=ENTRY

Function

Indicates initial entry into a menu. When the value of &$MHOPT is ENTRY, you can use the menu service procedure to perform any special processing that is required before the display of the menu.

Read-Only Variables

&$MHOPT
This variable is set to ENTRY.

&$MHAPPLID
This variable is set to the ID of the application to which the menu belongs.

&$MHMENUNUM
This variable is set to the menu number.

Modifiable Variables

&$MHAUTHMAP
Use this variable to control the access to menu options. This variable can be set to a map (15 characters in length) with each character corresponding to a menu option. Valid values for each character are:

1
Access to the menu option is allowed.

0
Access to the menu option is not allowed.

If this variable is set, it is used as the menu map for the user. If it is not set, access is allowed to all menu options.

&$MHINWAIT
Use this variable to set a timeout when the menu is not receiving any input. This variable can be set to an INWAIT value (between 0 and 86400 seconds) that is used when displaying the menu. If this value is zero, the menu does not time out. If the INWAIT time expires, the service procedure is called with &$MHOPT set to TIMEOUT.

&$MHDOUBLESPC
Use this variable to specify whether a blank line should be inserted before each input line on the menu. If this variable is set to YES (the default), a blank line is inserted before each input field line on the menu. If this variable is set to NO, input field lines are displayed as defined for the menu.

&$MHINPUTJUST

Use this variable to specify the value of the JUST keyword on the #FLD panel statement. Valid values are LEFT, RIGHT, ASIS, and CENTER. The default is LEFT.

&$MHINPUTMODE

Use this variable to specify the value of the MODE keyword on the #FLD panel statement. Valid values are SBCS and MIXED. The default is SBCS.

&$MHCOMMAND

Use this variable to enable the processing of commands that are not known to the system. If this value is YES, the service procedure is called with &$MHOPT set to COMMAND.

&$MHFK1..24

Use these variables to override the standard function key actions.

&$MHFKLAB1..24

Use these variables to override the standard function key labels.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the menu service procedure to one of the following return codes:

Code	Meaning
0	Proceed; display the menu
8	Terminate the menu

&$MHOPT=SELECT

Function

Indicates that the user has selected a menu option. When the value of &$MHOPT is SELECT, you can use the menu service procedure to perform any special processing that is required upon a menu option being selected.

Read-Only Variables

&$MHOPT
This variable is set to SELECT.

&$MHAPPLID
This variable is set to the ID of the application to which the menu belongs.

&$MHMENUNUM
This variable is set to the menu number.

&$MHSELECT
This variable is set to the option the user selects from the menu.

&$MHSELECTNUM
This variable is set to the relative numeric position of the selected option on the menu, as defined on the CAS : Menu Options panel (see Figure 7-3).

Modifiable Variables

&$MHERRLIST
Use this variable to return the names of the input fields that contain incorrect values to the system. This variable can be set to the input fields that are in error in the form *fieldname1, fieldname2...fieldnameN*. The fields are placed in error status on the panel, and the cursor is located in the first field that is in error.

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the menu service procedure to one of the following return codes:

Code	Meaning
0	Proceed; execute the action
8	Error; redisplay the menu

&$MHOPT=RETURN

Function

Indicates that the action for a selected menu option has completed. When the value of &$MHOPT is RETURN, you can use the menu service procedure to perform any special processing that is required on the completion of the action.

Read-Only Variables

&$MHOPT
The variable is set to RETURN.

&$MHAPPLID
This variable is set to the ID of the application to which the menu belongs.

&$MHMENUNUM
This variable is set to the menu number.

&$MHSELECT
This variable is set to the menu option for which the action has completed.

Modifiable Variables

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the menu service procedure to one of the following return codes:

Code	Meaning
0	Proceed; display the menu
1	Terminate the menu and return to the previous primary menu
4	If &SYSMSG is set, redisplay the menu with the value of &SYSMSG as the error message; if &SYSMSG is not set, terminate the menu and return to the previous primary menu

All other return codes are treated as errors, and &SYSMSG must be set. The menu is redisplayed with the value of &SYSMSG as the error message.

$MHOPT=EXIT

Function

Indicates that a menu is terminating. When the value of &$MHOPT is EXIT, you can use the menu service procedure to perform any special processing that is required on the termination of the menu.

Read-Only Variables

&$MHOPT
This variable is set to EXIT.

&$MHAPPLID
This variable is set to the ID of the application to which the menu belongs.

&$MHMENUNUM
This variable is set to the menu number.

Modifiable Variables

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the menu service procedure to one of the following return codes:

Code	Meaning
0	Proceed; exit from the menu
16	Do not exit; redisplay the menu

&$MHOPT=COMMAND

Function

Indicates that a command that is not known to the system has been issued from the menu. When the value of &$MHOPT is COMMAND, you can use the menu service procedure to perform any special processing that is required to process the command.

Read-Only Variables

&$MHOPT
This variable is set to COMMAND.

&$MHAPPLID
This variable is set to the ID of the application to which the menu belongs.

&$MHMENUNUM
This variable is set to the menu number.

&$MHCMD
This variable is set to the command.

&$MHCMDPARMS
This variable is set to the command parameters.

Modifiable Variables

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the menu service procedure to one of the following return codes:

Code	Meaning
0	Command processed successfully
4	Command in error; if &SYSMSG is not set, the error message MH0003 is displayed

&$MHOPT=TIMEOUT

Function

Indicates that the menu has timed out. When the value of &$MHOPT is TIMEOUT, you can use the menu service procedure to perform any special processing that is required when the menu times out.

Read-Only Variables

&$MHOPT
This variable is set to TIMEOUT.

&$MHAPPLID
This variable is set to the ID of the application to which the menu belongs.

&$MHMENUNUM
This variable is set to the menu number.

Modifiable Variables

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the menu service procedure to one of the following return codes:

Code	Meaning
0	Redisplay the menu
1	Terminate the menu, and return to the previous primary menu
4	Terminate the menu

20

List Service Procedure Interface

This chapter describes the list service procedure interface that is available for Common Application Services (CAS) lists.

About the List Service Procedure Interface

List service procedures are installation-written NCL procedures that are available for performing specialized processing on lists.

A list service procedure must be present in order for the CAS list component to work. You specify the list service procedure when you define a list.

The list service procedure is executed by CAS at seven processing points:

- On initial entry to the list—&$LHOPT=INIT
- To retrieve a list entry—&$LHOPT=GET
- To process an action against a selected entry—&$LHOPT=ACTION
- To process an add request—&$LHOPT=ADD
- To process the ALL command—&$LHOPT=ALL
- Prior to terminating the list—&$LHOPT=TERM
- To process a command not recognized by CAS—&$LHOPT=COMMAND

You write your own list service procedures by using the variables described in the following sections.

The variables can be divided into two groups as follows:

- The first group of variables provide information to the procedure and are *not* modifiable (read-only). In particular, you should check the value of &$LHOPT to determine the stage of list processing for you to implement the required special processing.
- The second group of variables are modifiable and enable the procedure to return information to the system. Some of these variables are set with values already, but the procedure can change those values.

You can use your own variables in the list exit procedure. However, do not use variable names that start with #LH. The list service procedure is called with NOSHRVARS=(#LH). Once you specify your own variables, these variables are persistent through the different stages of list processing. That is, if you specify &A = ***variable-value***, the value of this variable is available on each subsequent call to the procedure.

$LHOPT=ACTION

Function

Indicates that a list entry has been selected for actioning. When the value of &$LHOPT is ACTION, you can use the list service procedure to perform any special processing that is required to apply an action on the list entry.

Read-Only Variables

&$LHOPT
This variable is set to ACTION.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHSORT

This variable is set to the sort expression defined for the list.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

&$LHACTION

This variable is set to the mnemonic entered by the user next to the entry to be actioned.

&$LHENTID

This variable is set to the identifier of the entry selected by the user for actioning.

&$LHENTD

This variable is set to the data associated with the entry selected by the user for actioning.

Modifiable Variables

&$LHSPD*c*

These variables contain service procedure data as set by the caller of $CACALL; *c* is between 0 and 5 alphanumeric and/or national characters. These variables are never set or cleared by the system and must be completely managed by your installation-written NCL procedures.

&$LHREBUILD

Use this variable to indicate whether the entry line is to be rebuilt. This variable can be set to YES or NO. If set to YES, all the attributes to be used to rebuild the entry line must be set to the appropriate values. The default is NO. This variable is cleared by the system before calling the service procedure for ACTION processing.

&$LHENTMSG

Use this variable to set a message that is to overlay the entry line. The offset and length used are those defined in the list definition. This variable is cleared by the system before calling the service procedure for ACTION processing.

&$LHREFRESH

Use this variable to indicate whether the list is to be refreshed. This variable can be set to YES or NO. If set to YES, variables &$LHREBUILD and &$LHENTMSG are ignored. The default is NO. This variable is cleared by the system before calling the service procedure for ACTION processing.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list service procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	Request denied or error; redisplay the list with the mnemonic set in error
8	An error occurred; terminate the list

&$LHOPT=ADD

Function

Indicates that the user has executed the ADD command. When the value of &$LHOPT is ADD, you can use the list service procedure to perform any special processing that is required to process the ADD command.

Read-Only Variables

&$LHOPT
This variable is set to ADD.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHSORT

This variable is set to the sort expression defined for the list.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

Modifiable Variables

&$LHSPD*c*

These variables contain service procedure data as set by the caller of $CACALL; *c* is between 0 and 5 alphanumeric and/or national characters. These variables are never set or cleared by the system and must be completely managed by your installation-written NCL procedures.

&$LHREFRESH

Use this variable to indicate whether the list is to be refreshed. This variable can be set to YES or NO. The default is NO. This variable is cleared by the system before calling the service procedure for ADD processing.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list service procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	Request denied or error; redisplay the list with the error message
8	An error occurred; terminate the list

&$LHOPT=ALL

Function

Indicates that the user has executed the ALL command. When the value of &$LHOPT is ALL, you can use the list service procedure to perform any special processing that is required for ALL command processing.

Read-Only Variables

&$LHOPT
This variable is set to ALL.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHSORT

This variable is set to the sort expression defined for the list.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

&$LHACTION

This variable is set to the mnemonic entered by the user as the parameter on the ALL command.

Modifiable Variables

&$LHSPD*c*

These variables contain service procedure data as set by the caller of $CACALL; *c* is between 0 and 5 alphanumeric and/or national characters. These variables are never set or cleared by the system and must be completely managed by your installation-written NCL procedures.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list service procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	Request denied or error; redisplay the list with the error message
8	An error occurred; terminate the list

&$LHOPT=COMMAND

Function

Indicates that the user has executed a command not recognized by the system. When the value of &$LHOPT is COMMAND, you can use the list service procedure to perform any special processing that is required to process the command.

Read-Only Variables

&$LHOPT
This variable is set to COMMAND.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC
This variable is set to the description of the list.

&$LHDATASRC
This variable is set to the data source defined for the list.

&$LHCRITTOTAL
This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*
These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHSORT
This variable is set to the sort expression defined for the list.

&$LHFMTFLD*n*
These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*
These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

Modifiable Variables

&$LHSPD*c*
These variables contain service procedure data as set by the caller of $CACALL; *c* is between 0 and 5 alphanumeric and/or national characters. These variables are never set or cleared by the system and must be completely managed by your installation-written NCL procedures.

&$LHCOMMAND
This variable is set to the entered command. This variable can be set to a value that is to be displayed in the **Command** field on the list when &$LHSETCMDFLD is set to YES.

&$LHSETCMDFLD
Use this variable to indicate whether the value of &$LHCOMMAND is to be set in the **Command** field on the list when &RETCODE is set to 4. This variable can be set to YES or NO. The default is NO.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list service procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	Invalid command or error; redisplay the list with the error message
8	An error occurred; terminate the list

&$LHOPT=GET

Function

Indicates that a list entry is to be retrieved. When the value of &$LHOPT is GET, you can use the list service procedure to perform any special processing that is required to get a list entry.

Read-Only Variables

&$LHOPT
This variable is set to GET.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the displayed list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHSORT

This variable is set to the sort expression defined for the list.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

&$LHTIMEOUT

This variable is set to indicate whether the list panel timed out. This variable is set to one of the following values:

NO

The list panel did not time out.

YES

The list panel timed out.

&$LHRFRESHCMD

This variable is set to indicate whether the REFRESH command was executed by the user. This variable is set to one of the following values:

NO

The REFRESH command was not executed.

YES

The REFRESH command was executed.

&$LHDIRECTION

This variable is set to indicate the retrieval direction. This variable is set to one of the following values:

FWD

Get an entry in a forward direction.

BKWD

Get an entry in a backward direction.

LOCATE

Get the entry with an identifier that matches the value set in &$LHSKIP, if not found, get the first entry which has an identifier less than the value set in &$LHSKIP.

If the value of this variable is null, get the entry identified in &$LHENTID.

&$LHGETALL

This variable is set to indicate whether all entries for the list are retrieved during initialization processing for the list. This variable is set to one of the following values:

NO

Entries are retrieved as required for display on the list.

YES

All entries for the list are retrieved during initialization processing for the list.

&$LHSUPPRESS

This variable is set to indicate whether entries will be suppressed from the list by this procedure when getting entries, by the list exit procedure during entry processing, or by both procedures.

&$LHSKIP

This variable is set to indicate the number of entries to be skipped if &$LHDIRECTION is set to FWD or BKWD. If &$LHDIRECTION is set to LOCATE, it is set to the identifier of the entry to be located, that is, the locate string entered by the user.

&$LHGETFWD#

This variable is set to indicate the number of entries that are to be retrieved in a forward direction until the list is displayed.

Modifiable Variables

&$LHSPD*c*

These variables contain service procedure data as set by the caller of $CACALL; *c* is between 0 and 5 alphanumeric and/or national characters. These variables are never set or cleared by the system and must be completely managed by your installation-written NCL procedures.

&$LHENTTOTAL

Use this variable to specify the number of list entries to get for display. This variable can be set to the total number of entries to be displayed on the list when &$LHOPT is set to GET and &$LHRFRESHCMD or &$LHREFRESH is set to YES. The value of this variable is displayed in the top right-hand corner of the list.

&$LHENTID

This variable is set to the identifier of the entry from which to start reading. If &$LHDIRECTION and &$LHSKIP are null, get the entry that has an identifier that matches the value of this variable and if not found return NOT FOUND condition. If this variable is null and &$LHDIRECTION is set to FWD or BKWD, get the first or last entry respectively. You must set this variable to the identifier of the entry returned.

&$LHENTD

This variable is set to the data associated with the entry identified in &$LHENTID. This variable can be set to the data to be associated with the entry returned.

&$LHENTPOS

Use this variable to specify the position of the entry being returned within the list. If &$LHSUPPRESS is not set to YES and the Get all Entries field in the list definition is set to NO, the value of this variable is displayed in the top right-hand corner of the list in front of the total.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list service procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	Entry not found
8	An error occurred; terminate the list

&$LHOPT=INIT

Function

Indicates initialization of a list. When the value of &$LHOPT is INIT, you can use the list service procedure to perform any special processing that is required before the display of the list.

Read-Only Variables

&$LHOPT
This variable is set to INIT.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHSORT

This variable is set to the sort expression defined for the list.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

Modifiable Variables

&$LHSPD*c*

These variables contain service procedure data as set by the caller of $CACALL; *c* is between 0 and 5 alphanumeric and/or national characters. These variables are never set or cleared by the system and must be completely managed by your installation-written NCL procedures.

&$LHSUPPRESS

This variable is set to indicate whether entries will be suppressed from the list by this procedure when getting entries, by the list exit procedure during entry processing, or by both procedures. Set this variable to YES if you want to suppress entries; otherwise, set this variable to NO (the default).

&$LHADDALLOW

This variable is set to the Add Allowed setting defined for the list. It is set to YES if the list supports the ADD command, and it is set to NO if the list does not support the ADD command. The value of this variable can be modified to YES or NO. This variable is ignored if the list is a single or multiple select, or numbered list.

&$LHGETALL

This variable is set to the Get All Entries setting defined for the list. It is set to YES if all entries for the list are retrieved during initialization processing for the list. It is set to NO if entries are retrieved as required for display on the list. The value of this variable can be modified to YES or NO.

&$LHBUILDBKWD

This variable is set to indicate whether entries for the list can be built in a backwards direction when processing the BACKWARD command—this means that the entries are retrieved in a backwards direction. Set this variable to YES if entries can be built in a backwards direction; otherwise, set this variable to NO. The default value is YES.

&$LHACTIONS

This variable must be set to the mnemonics of the supported actions in the format *mmm,mmm,mmm....*

&$LHCONFIRM

This variable can be set to the mnemonic(s) of the action(s) that are to be confirmed before the action occurs, in the format *mmm,mmm,mmm....*

&$LHENTTOTAL

Use this variable to specify the total number of entries to be displayed on the list. The value of this variable is displayed in the top right-hand corner of the list if &$LHSUPPRESS is not set to YES.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list service procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	No entries found; terminate the list
8	An error occurred; terminate the list

&$LHOPT=TERM

Function

Indicates the termination of a list. When the value of &$LHOPT is TERM, you can use the list service procedure to perform any special processing that is required for terminating the list.

Read-Only Variables

&$LHOPT
This variable is set to TERM.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC
This variable is set to the description of the list.

&$LHDATASRC
This variable is set to the data source defined for the list.

&$LHCRITTOTAL
This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*
These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHFMTFLD*n*
These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHSORT
This variable is set to the sort expression defined for the list.

&$LHATB*
These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

Modifiable Variables

&$LHSPD*c*
These variables contain service procedure data as set by the caller of $CACALL; *c* is between 0 and 5 alphanumeric and/or national characters. These variables are never set or cleared by the system and must be completely managed by your installation-written NCL procedures.

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list service procedure to one of the following return codes:

Code	Meaning
0	Continue processing
8	An error occurred

21

List Exit Procedure Interface

This chapter describes the list exit procedure interface that is available for Common Application Services (CAS) lists.

About the List Exit Procedure Interface

List exit procedures are installation-written NCL procedures that are available for lists to perform specialized processing.

List exit procedures are optional. You specify the list exit procedure when you define a list. The list exit procedure is executed by CAS at three processing points: initialization, entry processing, and termination.

You write your own list exit procedures by using the variables described in the following sections.

The variables can be divided into two groups as follows:

- The first group of variables provide information to the procedure and are *not* modifiable. In particular, you should check the value of &$LHOPT to determine the stage of list processing for you to implement the required special processing.
- The second group of variables are modifiable and enable the procedure to return information to the system. Some of these variables are set with values already, but the procedure can change those values.

You can use your own variables in the list exit procedure. However, do not use variable names that start with #LH. The list exit procedure is called with NOSHRVARS=(#LH). Once you specify your own variables, these variables are persistent through the different stages of list processing. That is, if you specify &A = ***variable-value***, the value of this variable is available on each subsequent call to the procedure.

$LHOPT=INIT

Function

Indicates initialization of a list. When the value of &$LHOPT is INIT, you can use the list exit procedure to perform any special processing that is required to initialize the list.

Read-Only Variables

&$LHOPT
This variable is set to INIT.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

Modifiable Variables

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHSUPPRESS

Use this variable to specify whether entries are to be suppressed from the list by this procedure during entry processing. Set this variable to YES if you want to suppress entries; otherwise, set this variable to NO (the default).

&$LHADDALLOW

This variable is set to the Add Allowed setting defined for the list. It is set to YES if the list supports the ADD command, and it is set to NO if the list does not support the ADD command. The value of this variable can be modified to YES or NO. This variable is ignored if the list is a single or multiple select, or numbered list.

&$LHGETALL

This variable is set to the Get All Entries setting defined for the list. It is set to YES if all entries for the list are retrieved during initialization processing for the list. It is set to NO if entries are retrieved as required for display on the list. The value of this variable can be modified to YES or NO.

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
8	An error occurred; terminate the list

&$LHOPT=ENTRY

Function

Indicates entry processing. When the value of &$LHOPT is ENTRY, you can use the list exit procedure to perform any special entry processing that is required for the list.

Read-Only Variables

&$LHOPT
This variable is set to ENTRY.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

Modifiable Variables

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	Suppress the entry from the list
8	An error occurred; terminate the list

$LHOPT=TERM

Function

Indicates termination of a list. When the value of &$LHOPT is TERM, you can use the list exit procedure to perform any special processing that is required to terminate the list.

Read-Only Variables

&$LHOPT
This variable is set to TERM.

&$LHLISTFMT
This variable is set to indicate the format of the list. This variable is set to one of the following values:

ACTION
An action list.

MSELECT
A multiple select list.

SSELECT
A single select list.

NUMBERED
A numbered list.

&$LHAPPLID
This variable is set to the ID of the application to which the list belongs.

&$LHLISTTYPE
This variable is set to the type of the list. This variable is set to one of the following:

PUBLIC
The list is a public list.

PRIVATE
The list is a private list.

&$LHLISTUSER
This variable is set to the user ID of the user who owns the list, if it is a private list.

&$LHLISTNAME
This variable is set to the name of the list.

&$LHDESC

This variable is set to the description of the list.

&$LHDATASRC

This variable is set to the data source defined for the list.

&$LHCRITTOTAL

This variable is set to the number of &$LHCRIT variables that contain criteria. The value of this variable is in the range 0 to 9999.

&$LHCRIT*n*

These variables are set to the criteria which the service procedure uses to determine the entries to be included in the list, if variable &$LHCRITTOTAL is greater than zero; *n* is in the range 1 to the value of &$LHCRITTOTAL.

&$LHFMTFLD*n*

These variables are set to the names of the real fields defined for the list; *n* is in the range 1 to the number of real fields defined.

&$LHATB*

These variables are set to the panel attributes that can be used to set the intensity, color, and highlighting for data within an entry line. See Appendix E, *List Panel Attributes*, for a list of these variables.

Modifiable Variables

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the list exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
8	An error occurred

22

Criteria Exit Procedure Interface

This chapter describes the criteria exit procedure interface that is available for Common Application Services (CAS) criteria.

About Criteria Exit Procedure Interface

Criteria exit procedures are installation-written NCL procedures that are available for performing specialized processing when criteria are built.

Criteria exit procedures are optional. You specify the criteria exit procedure when you define a set of criteria. A criteria exit procedure performs installation-specific criteria processing at defined points during the building of a set of criteria:

- On initialization
- When a user performs one of the following from a run-time criteria panel:
  - Press ENTER
  - Issue the ACTION command
  - Issue the EXIT command

You write your own criteria exit procedures by using the variables described in the following sections.

The variables can be divided into two groups as follows:

- The first group of variables provide information to the procedure and are *not* modifiable (read-only). In particular, you should check the value of &$CROPT to determine the stage of criteria processing for you to implement the required special processing.
- The second group of variables are modifiable and enable the procedure to return information to the system.

You can use your own variables in the criteria exit procedure. However, do not use variable names that start with #CR. The criteria exit procedure is called with NOSHRVARS=(#CR). Once you specify your own variables, these variables are persistent through the different stages of criteria processing. That is, if you specify &A = ***variable-value***, the value of this variable is available on each subsequent call to the procedure.

&$CROPT=INIT

Function

Indicates initialization processing on the criteria. When the value of &$CROPT is INIT, you can use the criteria exit procedure to perform any special initialization processing on the criteria.

Read-Only Variables

&$CROPT
This variable is set to INIT.

&$CRAPPLID
This variable is set to the ID of the application to which the criteria set belongs.

&$CRCRITTYPE
This variable is set to one of the following types:

PUBLIC
The criteria set is public.

PRIVATE
The criteria set is private.

&$CRCRITUSER
If the criteria set is private, this variable is set to the ID of the user to whom the criteria set belongs. If the criteria set is public, this variable is not set.

&$CRCRITNAME
This variable is set to the name of the criteria.

&$CRDESC
This variable is set to the description of the criteria.

&$CRPANEL
This variable is set to the name of the run-time panel defined for the criteria.

&$CRDATASRC
This variable is set to the data source defined for the criteria.

&$CREPARMTOT
This variable is set to the number of exit parameter lines defined for the criteria in the range 0 to 9999.

&$CREPARM*n*

This range of variables are set to the exit parameters defined for the criteria if &$CREPARMTOT is greater than zero; *n* is in the range 1 to the value of &$CREPARMTOT.

&$CRFKA1

This variable is set to the first line of the function key area if a run-time panel is defined for the criteria. If a run-time panel is defined, this variable must be specified as an output variable on the first line of the #TRAILER statement in the panel definition.

&$CRFKA2

This variable is set to the second line of the function key area if a run-time panel is defined for the criteria. If a run-time panel is defined, this variable must be specified as an output variable on the second line of the #TRAILER statement in the panel definition.

Modifiable Variables

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the criteria exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
8	An error occurred; terminate building the criteria

&$CROPT=TERM

Function

Indicates one of the following occurred from a run-time criteria panel:

- ENTER key pressed
- ACTION command issued
- EXIT command issued

This call is not applicable if there is no run-time criteria panel. When the value of &$CROPT is TERM, you can use the criteria exit procedure to perform any special processing on the run-time criteria.

Read-Only Variables

&$CROPT
This variable is set to TERM.

&$CRAPPLID
This variable is set to the ID of the application to which the criteria set belongs.

&$CRCRITTYPE
This variable is set to one of the following types:

PUBLIC
The criteria set is public.

PRIVATE
The criteria set is private.

&$CRCRITUSER
If the criteria set is private, this variable is set to the ID of the user to whom the criteria belongs. If the criteria set is public, this variable is not set.

&$CRCRITNAME
This variable is set to the name of the criteria.

&$CRDESC
This variable is set to the description of the criteria.

&$CRPANEL
This variable is set to the name of the run-time panel defined for the criteria.

&$CRDATASRC
This variable is set to the data source defined for the criteria.

&$CREPARMTOT

This variable is set to the number of exit parameter lines defined for the criteria in the range 0 to 9999.

&$CREPARM*n*

This range of variables are set to the exit parameters defined for the criteria if &$CREPARMTOT is greater than zero; *n* is in the range 1 to the value of &$CREPARMTOT.

&$CRCURSOR

This variable is set to the current cursor position if a run-time panel is defined for the criteria. If a run-time panel is defined, this variable must be specified on the CURSOR parameter of the #OPT statement in the panel definition.

&$CRCOMMAND

This variable is set to the command executed from the run-time panel. It is null if the ENTER key was pressed and no command was entered. This variable must be used as the input field for the **Command** field on the run-time panel. This variable, when set, contains one of the following commands:

ACTION

The ACTION command was executed to allow the specification of variable data from the run-time panel to be included in the criteria.

EXIT

The EXIT command was executed to terminate the run-time panel.

Modifiable Variables

&$CRALARM

Use this variable to control the terminal alarm if a run-time panel is defined for the criteria. Valid values are YES to turn the alarm on and NO otherwise. This variable must be specified in the ALARM parameter of the #ERR statement in the panel definition.

&$CRERRFLDS

Use this variable to return the names of the input fields that contain incorrect values to the system. This variable can be set to the input fields that are in error in the form *fieldname1, fieldname2...fieldnameN.*

&SYSMSG

Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the criteria exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
4	Redisplay the run-time panel (not valid if &$CRCOMMAND is set to EXIT)
8	An error occurred; terminate building the criteria

23

Table Entry Validation Exit Procedure Interface

This chapter describes the table entry validation exit procedure interface that is available for Common Application Services (CAS) tables.

About Table Entry Validation Exit Procedure Interface

Table entry validation exit procedures are installation written NCL procedures that are available for performing specialized processing when a table entry is being maintained.

Table entry validation exit procedures are optional. You specify the table entry validation exit procedure when you define a table. The table entry validation exit procedure is called during table entry maintenance functions Add, Update, and Delete.

You write your own table entry validation exit procedures by using variables described in the following sections.

The variables can be divided into two groups as follows:

- The first group of variables provide information to the procedure and are *not* modifiable (read-only). In particular, you should check the value of &$VMEXFUNC to determine the operation being processed for you to implement the required special processing.
- The second group of variables are modifiable and enable the procedure to return information to the system. Some of these variables are set with values already, but the procedure can change those values.

&$VMEXFUNC=ADD

Function

Indicates that a table entry is being added. When the value of &$VMEXFUNC is ADD, you can use the table entry validation exit procedure to perform any special processing that is required when adding a table entry.

Read-Only Variables

&$VMEXFUNC
This variable is set to ADD.

&$VMEXAPPL
This variable is set to the application ID of the table definition.

&$VMEXFIELD
This variable is set to the field name of the table definition.

Modifiable Variables

&$VMEXFULL
This variable is set to the full value of the table entry.

&$VMEXABBR
This variable is set to the abbreviated value of the table entry.

&$VMEXDESC
This variable is set to the description of the table entry.

&$VMEXSEQ
This variable is set to the table entry sequence number.

&$VMEXACTIVE
This variable is set to indicate whether the table entry is active (YES or NO).

&$VMEXTXT1..10
These variables are set to additional information about the table entry.

&$VMEXERRFLD
Use this variable to identify the name of the variable in the table entry definition that is in error.

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the table entry validation exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
8	Do not complete the operation; redisplay the panel

&$VMEXFUNC=DELETE

Function

Indicates that a table entry is being deleted. When the value of &$VMEXFUNC is DELETE, you can use the table entry validation exit procedure to perform any special processing that is required when deleting a table entry.

Read-Only Variables

&$VMEXFUNC
This variable is set to DELETE.

&$VMEXAPPL
This variable is set to the application ID of the table definition.

&$VMEXFIELD
This variable is set to the field name of the table definition.

&$VMEXFULL
This variable is set to the full value of the table entry.

&$VMEXABBR
This variable is set to the abbreviated value of the table entry.

&$VMEXDESC
This variable is set to the description of the table entry.

&$VMEXSEQ
This variable is set to the table entry sequence number.

&$VMEXACTIVE
This variable is set to indicate whether the table entry is active (YES or NO).

&$VMEXTXT1..10
These variables are set to any additional information about the table entry.

Modifiable Variables

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the table entry validation exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
8	Do not complete the operation; redisplay the panel

&$VMEXFUNC=UPDATE

Function

Indicates that a table entry is being updated. When the value of &$VMEXFUNC is UPDATE, you can use the table entry validation exit procedure to perform any special processing that is required when updating a table entry.

Read-Only Variables

&$VMEXFUNC
This variable is set to UPDATE.

&$VMEXAPPL
This variable is set to the application ID of the table definition.

&$VMEXFIELD
This variable is set to the field name of the table definition.

&$VMEXFULL
This variable is set to the full value of the table entry.

Modifiable Variables

&$VMEXABBR
This variable is set to the abbreviated value of the table entry.

&$VMEXDESC
This variable is set to the description of the table entry.

&$VMEXSEQ
This variable is set to the table entry sequence number.

&$VMEXACTIVE
This variable is set to indicate whether the table entry is active (YES or NO).

&$VMEXTXT1..10
These variables are set to additional information about the table entry.

&$VMEXERRFLD
Use this variable to identify the name of the variable in the table entry definition that is in error.

&SYSMSG
Use this variable to return a message. If an error occurs during special processing, this variable must be set to an error message.

Return Codes

The variable &RETCODE must be set by the table entry validation exit procedure to one of the following return codes:

Code	Meaning
0	Continue processing
8	Do not complete the operation; redisplay the panel

Part V

Mapping Services

24

Mapping Services

This chapter describes the Mapping Services facility available in the SOLVE management services.

About Mapping Services

Mapping Services is a facility in the SOLVE management services that gives NCL access to complex data structures.

It allows NCL procedures to deal with the logical relationships and usage of data (the data *protocol*), whilst the system manages and maintains the physical representation of the data (the data *format*).

Abstract Syntax Notation One (ASN.1)

ISO 8824 defines the ASN.1 language as the standard mechanism for describing abstract data structures. The Mapping Services facility has adopted this language as the means to describe the logical (or abstract) data items within a data structure. The physical representation of the data can be defaulted from the ASN.1 definitions, or can be explicitly described by the inclusion of implementation specific definitions. This aspect is discussed in more detail later.

ASN.1 Type Assignments

ASN.1 defines data in terms of *types*. There are ASN.1 defined basic types that describe both *simple* data items, and *constructed* data items. By using these types as a foundation, ASN.1 allows the definition of user types. By combining data components of various types to form a new data type, complex data structures can be described.

The basic instrument of definition in the ASN.1 language is the *type assignment*. This allows the programmer to specify the name of a user type, and assign to that name one of the ASN.1 basic types (or perhaps another user defined type). If the type assigned is an ASN.1 constructed type, the definition is expanded to describe each of the components that comprise the structure. The constructed types are:

- SEQUENCE
- SEQUENCE OF
- SET
- SET OF

If a type is defined as a SEQUENCE (for example), then the components that comprise that sequence are listed as part of the type assignment. Each component in the constructed type is identified by a component name and its type.

A similarly useful type which is normally constructed is the type CHOICE which allows the definition of a number of alternate data components, one of which is chosen to complete an instance of data within a larger data structure.

As well as these five types there are a number of simple types, including INTEGER, BOOLEAN and various character string types. These are introduced later.

Type definitions can be reused to describe the underlying characteristics of different data items. For example, a data component named *errorCount* and another named *dayOfWeek* can both be defined as INTEGER, although they should not be confused by a programmer. Hence it is not the type that is significant in referring to the data items, but rather the component names.

Defining the Logical Structure of Data

Mapping Services uses ASN.1 as the source language for describing the logical structure of data. Any number of ASN.1 type assignments can be combined to form a map. Such a map is capable of describing a single data unit, or any number of differing data units. However the component that is associated with the map itself (and not structures defined within the map) must be defined as a single user type.

This is best explained by way of an example. Consider a Customer Order file that contains two types of record. One record type is called CustomerDetail and it contains customer specific information, such as name, address, contact, and so on. For each record of this type there are zero or more records of a type called OrderDetail, each of which contains specific order details related to the customer. The CustomerDetail and OrderDetail records are quite different so you could choose to use a separate map for each, but equally you can specify them in a single map as follows.

The map for the file is defined as being of type CustomerOrder. This is a user defined ASN.1 type, so a type assignment for CustomerOrder must form part of the source definition for the map. It need not be the first type assignment in the source definition but it is usually convenient in understanding the map definition if it is.

Since, in the above example, there are two types of possible record on the file, the CustomerOrder type assignment is best described by the ASN.1 type CHOICE. Its ASN.1 language definition might look like:

```
CustomerOrder ::= CHOICE {
      customer    CustomerDetail
      order       OrderDetail }
```

This means that the type CustomerOrder is a choice of either a component named *customer* which is of type CustomerDetail, or a component named *order* which is of type OrderDetail. Hence, although we have a map which consists of a single type, it immediately diverges into the two possible different record types that the file can contain. Subsequent type assignments specify the nature of CustomerDetail and OrderDetail. These are likely to be structures themselves, for example:

```
CustomerDetail ::= SEQUENCE {
      name        GraphicString
      address     Address
      openOrders  INTEGER }

Address ::= SEQUENCE {
      number      INTEGER
      street      GraphicString
      suburb      GraphicString }
```

and hence can contain either simple data items, such as *openOrders* and *name*, that are defined as the ASN.1 basic types INTEGER and GraphicString, or constructed data items such as *address*, which is of a user defined type Address. Eventually, through further type assignments, such as the one for the type Address shown, all structures are defined down to their elementary components.

Referencing Logical Data Structures From NCL

An NCL programmer can connect a data structure to a map in a number of ways. For example, during retrieval from a file:

```
&FILE GET MDO=NEXTREC MAP=CUSTOMER
```

The user supplies a local name to the MDO (in this case, NEXTREC) when reading it from the file, and nominates the map to be used to define the record contents (in this case, CUSTOMER). If you continue with the sample map described above, this record is defined as a CHOICE type. This means that you can refer to either NEXTREC.CUSTOMER, if the record is of type CustomerDetail, or NEXTREC.ORDER, if the record is of type OrderDetail.

You might know what type to expect (for example by understanding the key used to retrieve the record) and can immediately reference the inner structure. Otherwise, assuming that the records can be distinguished by Mapping Services due to some physical characteristic in the record itself, the component name of the choice within a particular record can be determined by using a query function.

In either case, reference to subsequent structures can proceed. The rules are simple. If a component is defined as a constructed type, (one of SEQUENCE, SEQUENCE OF, SET, SET OF, or CHOICE), then the one or more components within that construction are referenced by using the constructed component name, followed by a period, then the inner component name(s).

Hence in the example, the following logical components of a CustomerDetail record can be referenced from NCL:

```
NEXTREC.CUSTOMER.NAME

NEXTREC.CUSTOMER.ADDRESS.NUMBER

NEXTREC.CUSTOMER.ADDRESS.STREET

NEXTREC.CUSTOMER.ADDRESS.SUBURB

NEXTREC.CUSTOMER.OPENORDERS
```

Defining the Physical Structure of Data

A Mapped Data Object is simply a series of bytes. A subset sequence of these bytes can compose a logical structure. Some structures can be explicitly identified by the presence of enclosing lengths and some form of identifier that prefaces the data. Other logical structures might have no explicit identification but are defined only by external knowledge of the meaning of some sequence of bytes. Whatever the case, the physical representation of the data must be known to Mapping Services so that it can translate any NCL reference of a logical structure into the correct access technique corresponding to the physical data construction.

The system provides default rules for structuring data. If an MDO is created and always processed by NCL, you can choose to accept the default physical structure provided by the system. However, if an external data source is to be processed, or if a specific format needs to be devised for communication with some other program, then control of the physical representation is necessary.

Component Tags

Mapping Services allows the physical representation to be described by defining, for each unique component name referenced in a map, a unique *tag* (alternatively known as an identifier, or key), which is an integer that can be placed within the data to identify a logical component. If the component is constructed, the way in which its contents are represented can also be defined. This ensures that for a constructed component, each of the separate items within it can be isolated according to a common access technique and each can be independently identified by their unique tag values.

Local Form

The definitions required to support the physical representation of data are not part of the defined ASN.1 language. This is because ASN.1 does not define how data ought to be represented, but only its abstract syntax. An implementation of ASN.1 (such as SOLVE's Mapping Services) represents data in a *local form* that is convenient for subsequent processing in the local environment.

This allows the physical representation, and hence the local form, to be customized, and subsequently ensures that the data conforms to this defined view.

To allow the customization of local form data, the SOLVE ASN.1 Compiler recognizes a sequence of characters (which by definition are comments to ASN.1) and interprets the contents as implementation specific directives. It is these directives that can be used to instruct Mapping Services on the physical nature of the underlying data.

Data Interchange Between Open Systems

The other important aspect of ASN.1 is that the data it defines can be encoded for transmission. SOLVE supports the encoding and decoding of data using the Basic Encoding Rules (BER), ISO 8825. BER is specifically designed to interact with the logical structure as defined by ASN.1.

Independently of local form (that is, regardless of the physical representation of the data used for any MDO), the BER encoder can be used to build a datastream corresponding to the ASN.1 definition. This datastream can be understood by any BER decoder that has the same ASN.1 definition, even if it is a completely different ASN.1 platform.

For details on BER encoding and decoding in SOLVE refer to the NCL &ENCODE and &DECODE verb descriptions in the *Network Control Language Reference Manual*.

Map Source Definitions

The ASN.1 Language

In the following sections the way in which ASN.1 is implemented by SOLVE is explored, however it is not the intention to give a detailed explanation of all aspects of the ASN.1 language. You should refer to the official ISO documents for a complete understanding.

Only those aspects important and useful in understanding the mechanics of the Mapping Services implementation are discussed in depth.

Maps and ASN.1 Modules

A set of ASN.1 source definitions that is compilable is termed an ASN.1 *module*. ASN.1 modules can be registered in the SOLVE Map Library as a *map*. Modules are identified within the ASN.1 source, and this name must correspond to the registered map name. An ASN.1 module can also have a registered and unique identifier, called an *object identifier*, that officially identifies this module amongst all registered objects.

In general ASN.1 modules have a very free syntax, but Mapping Services imposes some restrictions on this for practical reasons.

Mapping Services Considerations

All names, both component and type identifiers, are restricted by Mapping Services to 32 characters in length. ASN.1 defines that a component name must start with a lowercase alphabetic character, and a type name with an uppercase alphabetic. Remaining characters can be chosen from the set of alphanumeric characters plus the hyphen. The Mapping Services compiler does not permit a hyphenated name, but instead allows the underscore (_) character.

The SOLVE ASN.1 compiler does not support ASN.1 macros, nor does it support complex subtyping. When introducing ASN.1 source containing macros or complex subtypes you must edit the source to resolve macros manually, and remove complex subtyping (possibly employing new code to perform value checks).

The SOLVE ASN.1 compiler does support simple subtyping such as sizes, integer and real value ranges and character set constraints.

The SOLVE ASN.1 Compiler ignores value assignments. They are checked by the compiler, but generate no output. However, named values appearing after a type assignment are supported.

Support is provided for the ASN.1 import facility, however imports are resolved during map load, and not during compilation. Any type definition can be imported from another module, provided it is marked for EXPORT within that module. The compiler produces an IMPORT list of types and the containing maps. All such maps must be compiled for the import to be successful on map load. Since importing takes place at map load time and not at compile time it is possible for a map load to fail due to incomplete or inconsistent definitions.

ASN.1 Module Layout

An ASN.1 module has a layout as depicted in the diagram shown below.

Note

Uppercase values are entered as shown. Lowercase values are expanded in the actual source. Angle brackets (<>) denote optional definitions and are not part of the source. The vertical bar (|) denotes alternatives and is not part of the source. Underlined values are defaults.

```
module identifier < module oid >

DEFINITIONS < EXPLICIT TAGS | IMPLICIT TAGS > :: =

BEGIN

< EXPORTS export list >
< IMPORTS import list >

< type assignments >
< value assignments >

END
```

ASN.1 Comments

Comments can be used anywhere within the module source. They begin with a sequence of two adjacent hyphens (--) and terminate with another pair, or at the end of the current line, for example:

```
-- this is an ASN.1 comment that ends here --

-- this is an ASN.1 comment that extends to the end of the line
```

Module Identifier

The module identifier names the module and can optionally provide the unique object identifier under which the module is registered.

Module Definitions

Following the DEFINITIONS keyword the default tagging options can be set to EXPLICIT TAGS (the default) or IMPLICIT TAGS. Tagging is discussed in more detail later. The assignment sequence (::=) is then followed by the BEGIN keyword. The BEGIN and END keywords bracket the module body.

Exports

The module body begins with an optional export sequence. This declares those definitions from within the module that are externally referencable. It takes the form:

```
EXPORTS symbol < ,symbol,symbol,...,symbol >
```

where each symbol is a *typeReference* or *valueReference* from the module body.

Imports

Exports are followed by an optional import sequence. This declares those external definitions that are required to complete the definitions within this module. It takes the form:

```
IMPORTS symbol < ,symbol,symbol,...,symbol > FROM module
    <  symbol < ,symbol,symbol,...,symbol > FROM module >
```

where symbol is a *typeReference* or *valueReference* defined in the identified module which is a module identifier.

Type Assignments

The main portion of the module body usually comprises a number of type assignments. Each type assignment has the form:

```
TypeReference ::= typeDefinition
```

where the *typeReference* is the name of a user defined type. The *typeDefinition* expands this user type according to any of the ASN.1 options which are discussed later.

Value Assignments

The module body can contain a number of value assignments. Each value assignment has the form:

```
valueReference type ::= valueDefinition
```

where the *valueReference* is the name of a user defined value, of the *type* referenced, and is assigned the *valueDefinition* referenced.

The SOLVE ASN.1 Compiler's Interpretation of the ASN.1 Module

Use of Comments

The compiler ignores ASN.1 comments that are not interpreted as implementation specific directives. It accepts Mapping Services directives, for manipulating the local form data, embedded in ASN.1 comments that start with the sequence (`--<`) and terminate with the sequence (`>--`). The Mapping Services directive start sequence must be explicitly terminated with the end sequence:

```
--< directive1,directive2,...,directiveN >--
```

The allowable directives, and where they can appear in the module source, are explained in more detail later.

Module Identifier

The ASN.1 module identifier is checked by the compiler to be the same as the map name that is being compiled. (This check is not case sensitive, only the uppercase version of the names must agree). If an object identifier is included, all integer values must be explicitly present for each portion of the registration arc.

Default Tagging

The compiler accepts the default tagging information as supplied.

Exports

The compiler accepts any EXPORT sequence. Only typeReferences and valueReferences named in the export sequence can be imported by other maps. Imports and exports are resolved at map load time.

Imports

The compiler accepts any IMPORT sequence, but the definitions are not imported during compilation. The list of imports are remembered in the compiled output, and the requested definitions imported only during map load in their compiled version. Hence compilation is performed without resolving that all type definitions exist for the module. However, the load fails unless resolution is complete following the import phase.

When a *typeReference* is imported, all contained component definitions are imported, plus all supporting typeReferences. Thus importing represents a convenient technique for rationalizing definitions such that several specific maps can include a set of common definitions. Import resolution is discussed further in the section dealing with map loading. Refer to the section titled *Loading a Map* in Chapter 25, *Maintaining Maps*. In the map being used for importing, the export sequence must contain the name of the Type Reference that is being imported.

Type Assignments

The compiler processes all type assignments, producing compiled output for each type and its constituent components. All ASN.1 base types are supported, except the EXTERNAL type. (If required, it can be defined as a user defined type). Each type is fully syntax checked, but cross-checking between individual components and their defined types is not performed. During map load the fully resolved map definitions are validated further to be logically consistent.

Value Assignments

The compiler accepts value assignment definitions but produces no output. These definitions are not used in NCL.

Map Components

Components are the data entities defined in a map that can be referenced by NCL. Every component has some data type associated with it. Data types give components their properties, but the types themselves are not referenced by NCL. A component's type can be a user defined type, which is defined by a type assignment within the source, or it could be a primitive ASN.1 type. Even where a component's type is a user defined one, by going to that type assignment (and possibly repeating this process), the type assignment specifies a primitive ASN.1 type. This primitive type is referred to as the *base type* for the component.

Component Definition

Although each component is of some type, the components themselves can only be defined within a type assignment that has a base type which is a *constructed type*, that is, one of SEQUENCE, SEQUENCE OF, SET, SET OF, or CHOICE.

Hence to define any components within a map at least one type assignment referencing one of the constructed types is required.

For a data component to be usable, ASN.1 only requires it to have a type. The compiler requires that the component is *named* (except in some circumstances discussed later). The component name must be unique (in its uppercase form) within the construct in which it is defined, and is the name used by NCL to reference that particular data entity.

The Mapped Data Object as a Component

When an NCL procedure creates an MDO, or receives one through an NCL verb, it connects explicitly or implicitly to a map. It is convenient to think of the MDO itself as a component. However, the MDO name is supplied by the user and is not defined within the map.

This component name (the MDO name) forms the first name segment when referencing any component defined within the MDO. As for other components, this first component is given a type, called the *map type*, which is the first *typeReference* encountered in the module body.

Component Name Hierarchy

When, as is typical, the map type is a constructed type, then it contains the definitions of one or more components that do (or can) comprise the data elements for that type. These components are referenced from NCL by concatenating each component name to the MDO name separated by a period, for example:

```
MDO=problem.number
```

where the MDO created by the user is the component *problem*, and the component *number* is defined within the component list for the map type. Likewise, if a named component itself has a type which is constructed, then the components defined within its type form the next level in the component name hierarchy.

Map Closure

A map is given a map type by the compile process. This is the first type definition encountered in the source. As described above, this type is typically a constructed type, containing the definitions of its contained components. Each of these components form the next level of the name hierarchy within the MDO, and if they are constructed, their components form the next level of the name hierarchy, and so on.

Overall, the relationship between components and their types maps out a complete hierarchical structure where the type names form the linkage between component names.

If a module contains type references which cannot be reached by following the type reference linkages, starting from the map type and working down, then these types are not usable within this map. Such isolated definitions might be the subject of an import statement from some other map, and hence such orphaned types are allowable. However, it is possible that they were intended to form part of the module in which they reside, and this is an indication that one of the following has occurred.

- The wrong type was selected as the map type (that is, a branch further down the tree than intended was the first type reference and has become the map type, leaving higher level types unreachable). This can be resolved by ensuring the correct type reference is the first in the module.
- The definitions as entered have more than one possible entry type (that is, there are a number of separate type and component hierarchies defined such that they do not completely overlap). This can be resolved by introducing a new superior type that contains a set of components that refer to each of the possible entry types (for example, by making it a CHOICE of these types).

This process is termed *map closure*, and ensures the completeness and consistency of the supporting map definitions for use by NCL.

Data Tagging

ASN.1 Tags

Maps can be developed from ASN.1 for the purpose of driving a transfer syntax, such as BER, in order to communicate in an open systems environment. The ASN.1 definitions describe how the data is serialized for transmission, and how each data component is *tagged* so that data items can be distinguished.

An ASN.1 tag consists of a tag class, and a tag number. There are four ASN.1 tag classes

- **UNIVERSAL** tags—are defined by ISO bodies.
- **APPLICATION** tags—are unique throughout an application.
- **PRIVATE** tags—are defined by private agreement.
- **CONTEXT-SPECIFIC** tags—have meaning in the immediate context only.

The tag number is an integer value greater than 0.

When an ASN.1 tag is encoded both the class and number are used to create a unique value such that the class and number remain separately decipherable.

Each ASN.1 base type has a predefined tag number in the UNIVERSAL tag class. Since all data components must be of a base ASN.1 type a default tag value exists for all data items. However ASN.1 allows these tags to be overridden and data to be explicitly tagged with any tag value. In fact, any data component can be tagged more than once, the tags being applied left to right in the order defined.

For example:

```
component1  [ APPLICATION 23 ]  [ 3 ]  INTEGER
```

produces 3 tags for *component1*,

- APPLICATION class, tag number 23
- CONTEXT-SPECIFIC (the default) class, tag number 3
- UNIVERSAL class, tag number 2 (for INTEGER)

In general, where a data component is explicitly tagged, it is followed by either more explicit tags, or its UNIVERSAL tag for the underlying base type as shown in the example above. However, use of the IMPLICIT keyword before a tag, or before a *typeReference*, indicates that the next tag is understood (implied), and is not encoded.

For example:

component1 [APPLICATION 23] IMPLICIT INTEGER

- produces just one tag, APPLICATION class, tag number 23

the INTEGER tag being implied and not encoded due to the presence of the IMPLICIT keyword.

Note that following the DEFINITIONS keyword in the module header the tagging defaults can be set for the module. The default is EXPLICIT TAGS indicating all tags are produced unless the IMPLICIT keyword is used to override. The IMPLICIT TAGS options can be used to indicate that only the first tag encountered for a component is used, the rest being implicit (except for the types CHOICE and ANY DEFINED BY, when all tags apply).

MDO Tags

In order to differentiate between data items maintained internally in their local form, Mapping Services also requires that all components have a recognizable tag.

Since a tag value prefixes most components in the MDO local form, the MDO is a self-defining structure which allows, for example, an MDO to be stored and later retrieved whilst the data within remains separately identifiable even where the map might have altered slightly. As long as map maintenance does not change the relationship between tag values and the components they represent, maps can be extended and modified without affecting the ability to process an MDO defined under a previous map version.

Mapping Services ignores tag classes, and just uses an integer value as a tag. However, the MDO tag numbers must be unique within a logical structure in the same way that component names must be unique within that structure. Hence within any structure (such as a sequence or a set) all components must have unique names (enforced by ASN.1) and corresponding to each component a unique MDO tag number (enforced by Mapping Services).

These MDO tags can be set explicitly and independently of ASN.1, or can be generated by the compiler. However the compiler can be directed to use the ASN.1 tag values as MDO tag values. This is likely to be the case where the map being developed has no requirement for BER encoding, hence all tag values are open to interpretation by Mapping Services only.

Note

By defining the tags explicitly in the source data the user is afforded a greater level of protection against change. MDOs that are created using one version of the map can often continue to be compatible with higher versions of the map, even though new components are added. However, if the compiler generates the tags this is unlikely to be the case.

Mapping Directives

MDO Tag Generation

Before the first ASN.1 statement in a module is encountered, one of the following directives can be used to indicate how MDO tags are to be generated.

```
--< TAGS(ANY) >--      compiler is to generate tags

--< TAGS(EXPLICIT) >-- tags are explicitly coded in the
source
```

ANY is the default, meaning that the compiler generates a unique tag value for each component reference in a module. Even where a component name occurs more than once in the module it is given a separate tag. In general this means that the actual tag values used are unpredictable across compiles. If there is no desire to control any MDO tag values, and no requirement to store MDOs in their local form, then this is the easiest way to define MDO tagging.

If EXPLICIT tagging is used, the compiler assumes that the first explicit ASN.1 tag found for a component is in fact the MDO tag value, regardless of its tag class. If no explicit tag is found for a component, the compiler generates a tag value for it as described above for TAG(ANY). Note that this means that the compiler does not use the default tagging for ASN.1 base types as they are likely to lead to ambiguities.

When defining explicit tags care must be taken to ensure that a tag value is used only once within a given structure. However, this tagging technique is very useful where control over tag values is desirable but there is no requirement to BER encode the data, and hence the tags used throughout the module need only be thought of as being MDO tags.

Alternatively, where it is necessary to BER encode the data, but it is also desired to control the MDO tags, the need might arise to use explicit MDO tags that are not part of the ASN.1 source. This can be done by using the following compiler directive following the component name being tagged:

```
--< [ nn ] >--
```

This directive follows the component name, but precedes any explicit ASN.1 tag or type information, for example:

```
userid        --< [ 10 ] >- [ PRIVATE 23 ] GraphicString
```

In this example, the *userid* component has an MDO tag value of 10, but if BER encoded, it is tagged with a private tag value of 23, and possibly the GraphicString universal tag, depending upon whether the tag option is set to IMPLICIT or EXPLICIT. The tag option immediately follows the word DEFINITIONS in an ASN.1 module.

Setting the MDO tag value this way always overrides any other MDO tag generation option.

Mapping Services is capable of maintaining MDOs according to a number of local form rules. By default it assumes all components are variable length items consisting of a tag (or key), a length, and the data itself. The following Mapping Services directives can be placed in the source to control the local form of data (*k* and *l* are integers in the range 0..4, where *k* is the length of the key, and *l* is the length of the length bytes):

```
--< KL0(k,l) >--      tag, length, then data, length is of
                      data only.

--< KL1(k,l) >--      tag, length, then data, length is of
                      length bytes + data.

--< KL2(k,l) >--      tag, length, then data, length is of
                      tag +length + data.

--< LK0(l,k) >--      length, tag, then data, length is of
                      data only.

--< LK1(l,k) >--    length, tag, then data, length is of tag
                    bytes + data.

--< LK2(l,k) >--      length, tag, then data, length is of
                      length + tag + data.
```

Such rules apply to structured components only. It is important to note that the rule applying to a structured component describes the manner in which its embedded components are managed and not the structure itself. The way in which the structure itself is managed depends upon the rule applying to its parent structure.

These directives can be used before any ASN.1 type assignment statement to set the compiler default for components encountered in the source module after that point, for example:

```
--< KL0(2,2) >--

CustomerOrder ::= CHOICE {
      customer    CustomerDetail
      order       OrderDetail }
```

In this example the components *customer* and *order* carry the KL0(2,2) encoding rule such that their embedded components (if any) are managed according to that rule.

This directive can also be applied to an individual component without changing the compiler default, for example:

```
--< KL0(2,2) >-

CustomerOrder ::= CHOICE {
      customer                  CustomerDetail
      order--< KL0(4,4) >--     OrderDetail }
```

In this example the *customer* component carries the compiler default KL0(2,2) encoding rule, while the *order* component carries the KL0(4,4) rule.

Note that in both the above examples, the *customer* and *order* components must be managed in the same manner. This is described by the encoding rule carried in the component which is of type CustomerOrder (not shown here).

Mapping Services can be used to map existing data structures and to accommodate this it can also manage components that do not have explicit length and tags in the data. Such data is considered fixed and fixed components can only be specified within a SEQUENCE or SEQUENCE OF type definition, for example:

```
CreateDetails ::= SEQUENCE {
      userid     --< FIX(8) >-- GraphicString,
      date       --< FIX(8) >-- GraphicString,
                                              --YY/MM/DD--
      time       --< FIX(8) >-- GraphicString}
                                              --HH:MM:SS--
```

In this example the *userid* component occupies the first 8 bytes of the data, the *date* component the next 8 bytes, and the *time* component the last 8 bytes. Although tags are not used within the data, each component can still be defined with a tag value, or the compiler generates one.

```
--< FIX(x,y) >-means the component is of length x, and starts
at offset y within its enclosing structure.  If y (offset)
is not specified, the offset is defaulted to the end of the
preceding component, or 0 (zero) if there is no preceding
component.
```

Default Formatting

All data types have both local form and external form defaults. Local form refers to the format that data is kept in within an MDO, while external form refers to the format of data as made available to, or supplied by, an NCL process. These default formats are described fully in the next section.

However, some types have alternative formats that can be set by compiler directives and are explained below.

Local Form for INTEGER

The local form of INTEGER types is, by default, a 1 to 4 byte signed binary field. However, IBM packed decimal and zone decimal formats can be managed by setting the options as one of the following (to request binary, packed decimal or zone decimal local form respectively):

```
INTEGER       --< BINARY >--

INTEGER       --< PACK >--

INTEGER       --< ZONE >--
```

External Form for REAL

The external form of REAL types is by default the NCL character representation of floating point numbers. However the external form can be modified by setting the number of integer digits and decimal places, for example:

```
REAL          --< EF(6,2) >--
```

This example requests up to 6 significant leading digits followed by 2 decimal places only.

Type Description and Formats

This section describes the types available within Mapping Services and their formats.

ASN.1 Types

There are a number of simple types defined by ASN.1, and most are supported by Mapping Services, as defined in the following list.

- BOOLEAN
- INTEGER
- BIT STRING
- OCTET STRING
- NULL
- OBJECT IDENTIFIER
- ObjectDescriptor
- REAL
- ENUMERATED
- NumericString
- PrintableString

- IA5String
- UTCTime
- GeneralizedTime
- GraphicString
- VisibleString
- GeneralString

One extension supported by Mapping Services is:

- HEX STRING

This is identical to the ASN.1 type OCTET STRING except in the way it is presented to NCL, as described in a later section. In addition, the following types are supported by the compiler, but have no real application to NCL (and are supported as if they are OCTET STRINGS):

- TeletexString
- VideotexString
- ANY and ADB

There are also a number of constructed types defined by ASN.1, and all are supported by Mapping Services:

- SET
- SET OF
- SEQUENCE
- SEQUENCE OF
- CHOICE

How Mapping Services implements each of these types is discussed below.

NCL Reference, Type Checking, and Data Behavior

When referencing an MDO in an NCL procedure, Mapping Services validates that the named component is defined (according to the name hierarchy supplied), and that the data within the component is valid, according to its underlying ASN.1 type. Each ASN.1 type can contain only certain valid values. Mapping Services checks the data value when retrieving data from, or assigning data into, an MDO. An operation attempting to retrieve or assign invalid data is rejected by Mapping Services with a message indicating a type check error.

In order to perform type checking Mapping Services first determines the base ASN.1 type of the component. Where a component is of a user defined type, the base ASN.1 type of the user defined type is inherited by the component. It is possible to have a number of levels of indirection between a user defined type and its base ASN.1 type.

The valid NCL values allowed for each of the base ASN.1 types is termed the *external* form. In addition to the set of valid values for each type, a specific component can be further constrained in what values are acceptable. Such *constraints* can be the result of either ASN.1 definitions or compiler directives. Finally, when data representing a valid NCL value is accepted for a component update, it is subject to a transformation from external form to *local form*, which is the MDO internal representation of data. This process carries with it further constraints.

The valid external form values, and the behavior of data managed by Mapping Services, is described for each type in the following sections.

The SET Type

Use

The SET type allows the definition of a fixed number of components where order is irrelevant. Within the set each component must be given a name, unless its base type is CHOICE. When a set is transferred from one system to another the items in the SET can be sent in any order. Items in the set can be optional, as indicated by the OPTIONAL keyword, for example:

```
Contact ::= SET {
      name     [1]      GraphicString,
      title    [2]      GeneralString,
      businessNumber    [3]   NumericString,
      afterHoursNumber  [4]   NumericString OPTIONAL}
```

When referring to SET items in NCL they are referenced by name, and hence the order that the data is stored in within the set is unimportant, and is in fact arbitrary.

An item in a SET can be a CHOICE of a number of alternatives. Such an item need not be explicitly named, but in this case each alternative of the choice must be named, and each must be unique amongst all other items within the entire SET. Alternatively, the SET item that is a CHOICE might indeed be named, and in this case only that name must be unique, as the CHOICE components are pushed to the next level in the component name hierarchy. For example, the following is valid:

```
Contact ::= SET {
      name          [4]   GraphicString,
      title         [5]   GeneralString,
      businessNumber      [6] NumericString,
      CHOICE {
            pagerNumber   [1] NumericString,
            homeNumber    [2] NumericString
             } ,
      additionalContact   [9] CHOICE {
                          pagerNumber[1]NumericString,
                          homeNumber[2]NumericString
                                    }
               }
```

This results in allowing the following component names to be referenced from NCL:

- Name
- Title
- BusinessNumber
- PagerNumber
- HomeNumber
- AdditionalContact.pagerNumber
- AdditionalContact.homeNumber

All SET items must be differentiated by an ASN.1 tag value. It is good practice to explicitly tag all SET items rather than default to tags of the ASN.1 base types.

External Form - Input and Output

All structures have an external form the same as their local form.

Local Form and Behavior

The local form of a SET is the datastream consisting of each set component, where each component has its leading tag and length (as determined by the parent component) and data in the local form for its type.

Named Values

Not applicable.

Constraints

Not applicable.

The SET OF Type

Use

The SET OF type allows the definition of either an arbitrary or fixed number of items of the same type, where order is not important. The definition includes only the type of the SET item, and does not name the component, for example:

```
PokerHand ::= SET OF Cards
```

In NCL, SET OF items are referenced by index only, within their parent structure. In the example above, the following index values are used to reference the set items of a component named *card* of type PokerHand:

```
card.{1}
card.{2}
...
card.{n}
```

External Form - Input and Output

All structures have an external form the same as their local form.

Local Form and Behavior

The local form of a SET OF type is the datastream consisting of each set component, where each component has its leading tag and length (as determined by the parent component) and data in the local form for its type.

Named Values

Not applicable.

Constraints

An upper limit can be placed on the number of items that the set can contain by use of the SIZE keyword. For example:

```
Pokerhand ::= SET SIZE(5) OF Cards
```

It is also possible to specify a SIZE range but the lower bound is ignored. For example, the following is equivalent to the above:

```
Pokerhand ::= SET SIZE(2..5) OF Cards
```

The SEQUENCE Type

Use

The SEQUENCE type allows the definition of a fixed number of components where order is relevant. As for a set, within a sequence each component must be given a name, unless its base type is a CHOICE. When a sequence is transferred from one system to another the items in the SEQUENCE are sent in the order defined. Items in the sequence can be optional, as indicated by the OPTIONAL keyword.

Mapping Services allows items of a SEQUENCE, or a SEQUENCE OF, construct (and only these constructs) to be of a fixed length. All other components must have a variable length structure, with a tag value so that each component can be recognized. Mapping Services ensures the sequence is reflected in the local form data. A compiler directive is used to indicate fixed data items, for example:

```
Contact ::= SEQUENCE {
       name   --< FIX(20) >--        GraphicString,
       birthDate --< FIX(8) >--             GraphicString,
                                     -- YY/MM/DD
       age    --< FIX(3) >--         INTEGER
       sex--  --< FIX(1) >--         ENUMERATED {
                                     female(0),
                                     male(1) },
       address              [1]     Address,
       previousAddress      [2]     Address }
```

In this example the first four components are fixed, and occupy the first 32 bytes of the SEQUENCE. The last two component are variable structures, and follow in sequence after the first 32 bytes.

External Form - Input and Output

All structures have an external form the same as their local form.

Local Form and Behavior

The local form of a SEQUENCE is the datastream consisting of each sequence component, where each component has its leading tag and length (where applicable, as determined by the parent component) and data in the local form for its type.

Named Values

Not applicable.

Constraints

Not applicable.

The SEQUENCE OF Type

Use

The SEQUENCE OF type allows the definition of either an arbitrary or fixed number of items of the same type, where order is important. The definition includes only the type of the SEQUENCE item, and does not name the component, for example:

```
Counters ::= SEQUENCE OF INTEGER
```

In NCL, SEQUENCE OF items are referenced by index only, within their parent structure. In the example above, the following index values are used to reference the sequence items of a component named *counters* of type Counters:

```
counter.{1}
counter.{2}
...
counter.{n}
```

External Form - Input and Output

All structures have an external form the same as their local form.

Local Form and Behavior

The local form of a SEQUENCE OF type is the datastream consisting of each set component, where each component has its leading tag and length (where applicable, as determined by the parent component) and data in the local form for its type.

Named Values

Not applicable.

Constraints

An upper limit can be placed on the number of items that the sequence can contain by use of the SIZE keyword. For example:

```
Counters ::= SEQUENCE SIZE(50) OF INTEGER
```

It is also possible to specify a range, but the lower bound is ignored.

The CHOICE Type

Use

The CHOICE type allows the definition of a number of components, each of which is an alternative in the data structure. A CHOICE component can be named, such as the *details* component in this example:

```
Person ::= SET {
      name        GraphicString,
      birthDate         GeneralString, -- YY/MM/DD
      age         INTEGER,
      details     CHOICE {
                  femaleInfo  [1]   FemaleInfo,
                  maleInfo    [2]   MaleInfo } }
```

The *details* component can contain either *femaleInfo* or *maleInfo*, but not both. In this case the *details* component containing the choice can be referenced directly without first knowing which choice was made. If this is changed to the following:

```
Person ::= SET {
      name  GraphicString,
      birthDate   GeneralString, -- YY/MM/DD
      age   INTEGER,
      CHOICE {
            femaleInfo  [1] FemaleInfo,
            maleInfo    [2] MaleInfo } }
```

then both *femaleInfo* and *maleInfo* are alternatives within the SET type, and must have unique names within that set. In this case, the actual choice that is present can only be discovered by attempting to reference each possible one, and determining whether it exists or not.

All alternatives of a CHOICE must have unique tags so that they can be differentiated. It is good practice to use explicit tags for each CHOICE item.

External Form - Input and Output

As for the choice item

Local Form and Behavior

As for the choice item.

Named Values

Not applicable.

Constraints

Not applicable.

The BOOLEAN Type

Use

The BOOLEAN type is used to represent a value of *true* or *false* only.

External Form - Input

The local character strings TRUE and FALSE (not case sensitive) are accepted, whilst the digit 0 is interpreted as false, and the digit 1 is true.

External Form - Output

The digit 0 (false) or 1 (true) is always returned.

Local Form and Behavior

Internally, Mapping Services stores a value of X'00' for false, and X'01' for true (and accepts any value other than X'00' as true).

For an input operation, where the component is variable length its length is always set to 1. Where the component length is fixed and is greater than 1 the value occupies the first byte only (it is left aligned) and the remainder of the component's data is set to zeros.

For an output operation, where the component is located and has a length greater than 1, only the first byte is inspected as the value.

Named Values

Not applicable.

Constraints

Not applicable.

The INTEGER Type

Use

The INTEGER type is used to contain any positive or negative whole numbers in the range -2,147,483,648 to 2,147,483,647 (that is, a signed 32 bit number).

External Form - Input

Valid input consists of a string of up to 15 digits optionally preceded by a plus sign (+) or a minus sign (-), which provides the sign (positive or negative) of the value. The sign is positive, if omitted. All other characters must be valid digits (that is, 0..9). Alternatively, if the map definition included named values for this component, the symbolic name of the named value can be supplied as external form input.

External Form - Output

Output consists of a string of one or more local characters. If the integer value is negative, the first character is a minus sign (-), otherwise the sign is omitted. All other characters are numeric characters. Leading zeroes are stripped.

Local Form and Behavior

Internally, Mapping Services can store integers in one of 3 formats:

Binary

Can be up to 4 bytes in length. If the length is not fixed, the value is kept in the smallest number of bytes possible. If the length is fixed, the value is right aligned and sign extended to the left.

Packed

Can be up to 8 bytes in length. The integer value is converted to the packed decimal equivalent. If the length is not fixed, the value is kept in the smallest number of bytes possible. If the length is fixed, the value is right aligned and zero padded to the left.

Zoned

Can be up to 15 bytes in length. The integer value is converted to the packed decimal equivalent. If the length is not fixed, the value is kept in the smallest number of bytes possible. If the length is fixed, the value is right aligned and zero padded to the left.

For any format, if a value exceeds that which can be stored without loss of significance a type check results. If a named value is input, the map definition is used to determine the actual integer value.

Named Values

It is possible to specify a range of names that correspond to particular integer values in an INTEGER type definition. For example:

```
x INTEGER      {red (1),
                green (5)
                blue (7)}
```

These names can then be used as external input to represent the corresponding value. On retrieval the integer value is returned.

Constraints

It is possible to constrain the allowed set of integers for an integer type. This is done by specifying a list of integer ranges and / or single values. For example:

```
( 1..10 | 20..30 | 50 | 100 )
```

In the above example the integer values are restricted to numbers 50, 100 and the range 1 to 30. If anything else is entered, a type check error results.

The BIT STRING Type

Use

The BIT STRING type is used to contain any data where individual bit values might have meaning. Mapping Services supports two types of BIT STRING access, Standard and Boolean. These are described below.

Standard BIT STRING Access

Use

Standard BIT STRING access deals with the string as a whole, allowing manipulation of the entire component through a single operation, as for most other types.

External Form - Input

Valid external form can be a string of one or more digits, each a zero (0) or a one (1). However, where named values are defined for the BIT STRING type, a list of named values, each separated by a plus sign (+) or a minus sign (-) sign, is an acceptable alternative. A named value preceded by a plus sign indicates that the named bit value should be set to true (the bit is set to 1), and a named value preceded by a minus sign indicates that the named bit value should be set to false (the bit is set to 0).

External Form - Output

The output format depends upon whether or not named values are defined for the BIT STRING type. Where no named values are defined the output consists of a string of zero or more (always a multiple of 8) digits, each a 0 or 1. Where one or more named values do exist the output is a character string comprising each named value where the named bit is 1 (meaning the value is true). Each name is delimited with a plus sign (+).

Local Form and Behavior

When a string of 0's and 1's is supplied as input, each digit in the input sequence is treated (left to right) as the value of the corresponding bit in that position of the local form data. If the number of bits supplied is not a multiple of 8, trailing bits are set to zero and padded to a byte boundary. If the component has a fixed length exceeding that of the input string, the value is left aligned, and all unreferenced bytes are set to X'00'. If the component cannot contain the number of input bytes supplied, the string is truncated.

When a list of named values, each preceded by a plus sign (+) or a minus sign (-), is supplied as input, only the named bits take part in the operation. Each named bit preceded by a plus sign (+) is set to 1 (true), whilst each named bit preceded by a minus sign (-), is set to 0 (false). All other bits in the BIT STRING are unaffected by the input operation.

When fetching the value of a BIT STRING a named value list is always returned if any named values are defined for the type, else a string of 0s and 1s is returned corresponding to the BIT STRING values. Note that when named values are defined all other bits in the BIT STRING are ignored on output regardless of their value.

Constraints

It is possible to specify a size constraint on a bit string type using the SIZE keyword. For example:

```
BIT STRING (SIZE(2.4))
```

The size refers to the maximum number of bits (not bytes) in the bit string. Internally, the minimum is rounded down, and the maximum rounded up, to the nearest multiple of 8.

Named Values

It is possible to specify a list of names corresponding to particular bit positions, starting from zero. For example:

```
{ FLAG1(0), ALLSET (1), ...}
```

It is possible to use a list of these names, preceded by a plus or minus signs, to indicate set or not set as external form input for the bit string component. When output the names of the set bits, separated by plus signs are returned, for example:

```
+FLAG1+ALLSET ...
```

Use

Boolean BIT STRING access deals with individual bit level access and operates only through named values. This access is recommended because program access to bits is only via their symbolic named values, thus removing from NCL the need to know relative bit positions.

For Boolean BIT STRING access to be invoked the named value of a bit is provided by NCL as an additional name segment after the BIT STRING component name. Since the BIT STRING type is primitive, the additional name in the name hierarchy is understood to be a named value, and is treated as a BOOLEAN type. No matter where the named value is in the BIT STRING the value of the bit is always 0 or 1, as for a BOOLEAN type.

External Form - Input

The local character strings TRUE and FALSE (not case sensitive) are accepted, whilst the digit 0 is interpreted as false, and the digit 1 is true.

External Form - Output

The digit 0 (false) or 1 (true) is always returned.

Local Form and Behavior

The component name plus the named value is treated as a reference to a specific bit (the bit position within the component being defined by the named value), and that bit is set to 0 or 1 depending upon the input. No other bits in the BIT STRING component are affected. If the component is extended to accommodate the input, all other bits are set to 0.

Named Values

Not applicable.

Constraints

Not applicable.

The OCTET STRING Type

Use

The OCTET STRING type is used to contain any data where no formatting is required.

External Form - Input

Any data.

External Form - Output

Data is returned unchanged.

Local Form and Behavior

Data is stored as is. If the component has a fixed length exceeding that of the input string, the data is left aligned, and all unreferenced bytes are set to X'00'. If the component cannot contain the number of input bytes supplied, the string is truncated.

Named Values

Not applicable.

Constraints

The SIZE keyword can be used to constrain the length of an octet string to a certain range or value. Length is measured in bytes. For example:

```
OCTET STRING (SIZE(4..8))
```

If the component is variable length, a type check error occurs if the size constraints are breached.

The HEX STRING Type

Use

The HEX STRING type is an Mapping Services extension to ASN.1, but is processed as a base ASN.1 type. It is identical in all respects to the ASN.1 OCTET STRING type except for its external form representation.

External Form - Input

Valid input consists of a string of one or more local characters, each selected from the set 0123456789ABCDEF. Each pair of hexadecimal characters represents a single byte value. If an odd number of characters is supplied, the string is treated as though padded on the left with a single zero (0).

External Form - Output

Data is returned in hexadecimal characters, as for input. An even number of characters is always returned.

Local Form and Behavior

Each two hexadecimal characters of input represents the actual data to be stored in a single byte. Otherwise behavior is as for OCTET STRING.

Named Values

Not applicable.

Constraints

The SIZE keyword can be used to limit the size of a HEX STRING type. The size refers to local form, not external form, length.

The NULL Type

Use

The NULL type is used where data in a component either must be null (that is, empty), or not accessible.

External Form - Input

The only valid input is a null value.

External Form - Output

A null value is always returned.

Local Form and Behavior

The component can be created by an input operation, but no contents are modified. If it already exists, no data is modified.

Named Values

Not applicable.

Constraints

Not applicable.

The OBJECT IDENTIFIER Type

Use

The OBJECT IDENTIFIER type is used to contain object identifier values that uniquely identify registered objects.

External Form - Input

Any sequence of characters from the set 0123456789 and the period (.), is valid provided it does not begin or end with a period (.), contains no consecutive periods, and contains at least one period. Each sequence of decimal digits punctuated by a period (.), represents a sub-identifier in the series of sub-identifiers that comprise an object identifier value.

External Form - Output

As for input.

Local Form and Behavior

The data format is as supplied for input, however truncation is not allowed. If the component is fixed length, it must be able to contain the input string, otherwise a type check results. If necessary, it is padded with blanks.

Named Values

Not applicable.

Constraints

Not applicable.

The ObjectDescriptor Type

Use

The ObjectDescriptor type is used to contain object descriptions for registered objects.

External Form - Input

As supplied.

External Form - Output

As supplied.

Local Form and Behavior

The data format is as supplied for input. If the component is fixed length, a short input string is padded with blanks, and a long input string is truncated.

Named Values

Not applicable.

Constraints

A character set constraint can be specified for ObjectDescriptor types. In addition, a size constraint can be specified. See the section *The GraphicString Type* for details.

The EXTERNAL Type

The EXTERNAL type is not supported by Mapping Services. If required, define it as a user defined type.

The REAL Type

Use

The REAL type is used to contain floating point, or scientific notation, numbers in the range 10^{-70} to 10^{70}.

External Form - Input

Format allowed is:

+ or -	(optional, plus or minus sign, followed by)
nnnnnn	(optional, any number of digits, followed by)
.	(optional, decimal place, followed by)
mmmmmm	(optional, any number of digits, followed by)
E*sxx*	(optional, signed exponent power of 10, range -99 to 99)

such that either, or both, *nnnnnn* and *mmmmmm* are present, and the resulting REAL number is within the allowable range.

Valid examples:

```
1457892345509676544283940 4
```

```
-123.567
```

```
.555
```

```
.0023E-23
```

```
3.142776589E+66
```

External Form - Output

The default is a normalized decimal real number:

+ or -	(plus or minus sign of the value, followed by)
.	(decimal place indicator, followed by)
nnnnnn	(15 significant fraction digits, followed by)
E*sxx*	(signed exponent power of 10)

Examples:

```
+.314277658900000E-10
```

```
-.123456789000000E+52
```

An MS compiler directive is available to specify how a REAL type is to be presented externally. For example:

```
x REAL --< EF(3,5)>--
```

The above directive gives an external form for the number

```
+.314277658900000E-10 as:

  3.14277
```

This has three digits before the decimal point, (including two blanks) and five digits after the decimal point.

Space padding occurs on the left if required, and zero padding occurs on the right. No truncation takes place. The external form expands to allow all of the digits that precede the decimal point in a real number to be represented in full.

Local Form and Behavior

For IBM S/370 machines, local form is a 64-bit *long floating point* value, and the component must be at least 8 bytes in length. Truncation is not allowed. If the component has a fixed length greater than 8 bytes, the value is left aligned and padded on the right with zero bytes.

Named Values

Not applicable.

Constraints

Real types can be restricted to a set of real value ranges or simple real values. For example:

```
REAL ({150, 10, -2}..{150, 10, -1})
```

The above example restricts the real type to the range (1.5...15). It is possible to specify a real range using whole numbers as well, for example:

```
REAL (1..20)
```

The ENUMERATED Type

Use

The ENUMERATED type is used to constrain a component to a defined set of values. Each defined value is *named* using a name identifier similar to a component name. Associated with each name is a unique integer value (which can be signed), for example:

```
Color  ::= ENUMERATED { red(0),blue(1),yellow(2),
                        green(3),black(7) }
```

External Form - Input

The external form must be one of the names listed in the ENUMERATED type. The enumerated values are not allowed (that is, *red* is valid, *0* is not).

External Form - Output

Same as input.

Local Form and Behavior

Internally, the ENUMERATED value is kept in the same manner, and is subject to the same local form constraints, as an INTEGER of the binary local form.

Named Values

Not applicable.

Constraints

If a component indirectly references an enumerated type, it is possible to constrain it to a subset of the set of named values. For example:

```
X Y(ONE, FIVE, SIX)

Y ::= ENUMERATED {ONE(1), TWO(2), THREE (3), FOUR (4),
FIVE (5), SIX (6)}
```

Component X is restricted to subset (one, five, six) of named values of type Y (one, two, three, four, five, six).

The NumericString Type

Use

The NumericString is a subset of GeneralString which comprises:

0 to 9	(numeric characters)
	(the space, or blank character)

External Form - Input

Any string of valid characters as described above.

External Form - Output

Same as input.

Local Form and Behavior

On input, data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks, and a long input string is truncated.

Named Values

Not applicable.

Constraints

As for GraphicString.

The PrintableString Type

Use

The PrintableString is a subset of GeneralString which comprises:

a to z	(lowercase alphabetic characters)
A to Z	(uppercase alphabetic characters)
0 to 9	(numeric characters)
	(the space, or blank character)

’ ( ) + , - . / : = ?

External Form - Input

Any string of valid characters as described above.

External Form - Output
Same as input.

Local Form and Behavior
On input data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks, and a long input string is truncated.

Named Values
Not applicable.

Constraints
As for GraphicString.

The TeletexString Type

Use
None in NCL.

External Form - Input
As supplied.

External Form - Output
As supplied.

Local Form and Behavior
As for OCTET STRING.

Named Values
Not applicable.

Constraints
As for GraphicString.

The VideotexString Type

Use
None in NCL.

External Form - Input
As supplied.

External Form - Output
As supplied.

Local Form and Behavior
As for OCTET STRING.

Named Values
Not applicable.

Constraints

As for GraphicString.

The IA5String Type

Use

Transparent general character set (VisibleString plus control characters).

External Form - Input

As supplied.

External Form - Output

As supplied.

Local Form and Behavior

On input data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks, and a long input string is truncated.

Named Values

Not applicable.

Constraints

As for GraphicString.

The UTCTime Type

Use

Date and time, as Universal Coordinated Time (year without century numbers), in the format:

YYMMDDHHMM[*SS*]Z

GMT date and time (to minutes or seconds); Z indicates GMT time

YYMMDDHHMM[*SS*]*s*HHMM

local date and time (to minutes or seconds); with signed zone offset from GMT time (*s* = + or -)

External Form - Input

Any valid input as shown above.

External Form - Output

Any valid data as shown above.

Local Form and Behavior

On input data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks. Truncation is not allowed.

Named Values

Not applicable.

Constraints

Not applicable.

The GeneralizedTime Type

Use

Date and time, as GeneralizedTime (year includes century numbers), in the format:

YYYYMMDDHH[*MM*[*SS*]][*.f*]Z

GMT date and time (to hours, minutes or seconds); with optional fractional time units to any significance (hours, minutes or seconds); Z indicates GMT time

YYYYMMDDHH[*MM*[*SS*]][*.f*][*s*HHMM]

local date and time (to hours, minutes or seconds); with optional fractional time units to any significance (hours, minutes or seconds); with signed zone offset from GMT time (*s* = + or -)

External Form - Input

Any valid input as shown above.

External Form - Output

Any valid data as shown above.

Local Form and Behavior

On input data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks. Truncation is not allowed.

Named Values

Not applicable.

Constraints

Not applicable.

The GraphicString Type

Use

Transparent character set of graphic characters only.

External Form - Input

As supplied.

External Form - Output

As supplied.

Local Form and Behavior

On input data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks, and a long input string is truncated.

Named Values

Not applicable.

Constraints

There are three ways of constraining the GraphicString type:

Size constraint — the SIZE keyword can be used to constrain the number of bytes for the GraphicString type to a range of values or a single value. For example:

```
GRAPHICSTRING (SIZE(3..8))
GRAPHICSTRING (SIZE(5))
```

Character set constraint — you can constrain the valid characters that are allowed to be used on a GraphicString type as external form data by using the FROM keyword to specify a character set. For example:

```
GRAPHICSTRING (FROM("A"|"B"|"C"|"a"))
```

In the above example only strings consisting of the letters specified are treated as valid external input.

Character string constraint — a set of valid character strings can be specified for a graphic string type. For example:

```
GRAPHICSTRING ("ABC"|"PaR"c|"xYz"m)
```

The letter *m* can be placed after the string to indicate that case is irrelevant. The letter *c* (the default) is used to show case is relevant. If the case in the external input does not match the string, the input is rejected.

The VisibleString Type

Use

Transparent character set of graphic characters only.

External Form - Input

As supplied.

External Form - Output

As supplied.

Local Form

On input data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks, and a long input string is truncated.

Named Values

Not applicable.

Constraints
See GraphicString.

The GeneralString Type

Use
Transparent character set of both graphic and control characters.

External Form - Input
As supplied.

External Form - Output
As supplied.

Local Form
On input data is stored as supplied. If the component has a fixed length, a short input string is padded with blanks, and a long input string is truncated.

Named Values
Not applicable.

Constraints
See GraphicString.

The ANY and ADB Types

Use
Can contain any sort of data. Type can be changed dynamically by attaching a map at run time.

External Form
As for OCTET STRING.

Local Form and Behavior
As for OCTET STRING.

Named Values
Not applicable.

Constraints
Not applicable.

25

Maintaining Maps

This chapter describes the facilities for creating, maintaining and compiling map definitions for ASN.1 source code.

About Maps

Maps comprise a number of source statements, and are compiled to a loadable form. Map source can be kept in any convenient source library in a similar fashion to NCL procedures.

All compiled maps are lodged in a file that serves as the Map Library. From this library the loadable form of a map can be accessed on demand. Normally a map is loaded only when specifically requested for use by an NCL procedure. When no longer required it may be deleted from storage by the system.

The SOLVE Map Library Structure

The SOLVE Map Library is a keyed file that contains:

- Map registration records
- ASN.1 source records
- Load module records

For each map defined to the system there is one map registration record. This registration record contains global information that registers the unique map name, and other identification data. It also contains information about the system dataset where the source statement file can be located.

One or more load module records exist for each map. These records are compiled from the map source statements, and represent the amalgamation of the logical and physical data structure information into a form understandable by the SOLVE Map loader.

The source records contain the ASN.1 source code from the last successful compile of the map.

Creating and Maintaining the Map Source

A map must first be registered to the system before it can be used by Mapping Services. The process of registration defines some of the global attributes of the map that allow it to co-exist as a unique entity in the system.

Before adding maps to the SOLVE Map Library the user should be aware of the restrictions placed on the ASN.1 language by the Mapping Services implementation, as described in the previous chapter.

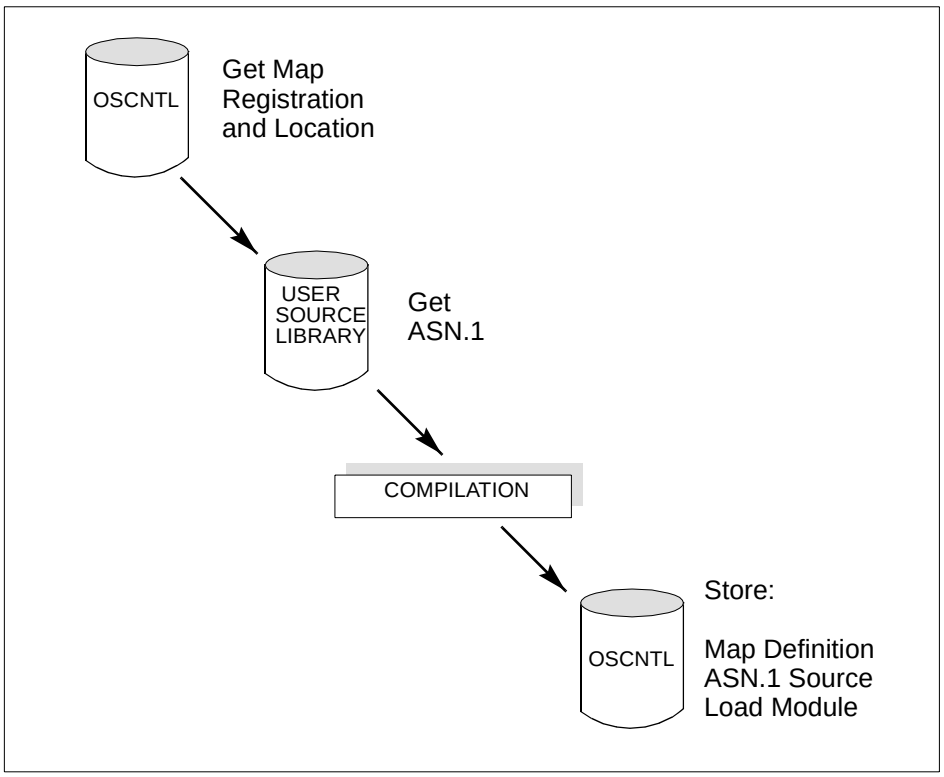
Compiling a Map

Once a map is registered and its ASN.1 source location defined the map can be compiled to produce the map load module. This module is a series of records kept in the Map Library which are accessed by the Mapping Services Loader when a map is loaded for use in the system.

If the compilation is successful, the source is stored in the Map Library for reference purposes.

The process of registering and compiling a map is illustrated in Figure 25-1.

Figure 25-1. Registering and Compiling a Map



Loading a Map

During the load phase the contents of the load module records are reduced to an internal format that can be used by Mapping Services. Since references to imported definitions are resolved on load, the load module containing the imported definitions must be previously compiled.

Any inconsistencies in either the ASN.1 logical structuring, or the Mapping Services local form definitions, causes the map load to fail.

Once compiled, maps can be loaded from the SOLVE Map Library to validate that they load without problems. Maps are normally loaded automatically when they are first referenced, requiring no action on your part. You might, however, want to load a map, using the SYSPARMS MAPLOAD command, to determine if it can be loaded successfully.

Defining a Map

You can define a new map by selecting the appropriate option from the Mapping Services Primary Menu or from a list of existing map definitions (by pressing the ADD key). You can also create a new map by copying an existing map definition. See *Copying a Map Definition*, on page 25-11.

Mapping Services Primary Menu

To access the Mapping Services : Primary Menu, select option **M** from the MODS : Primary Menu. This panel allows you to maintain map definitions, and to compile ASN.1 source code for the map.

The Mapping Services : Primary Menu is illustrated in Figure 25-2.

Figure 25-2. Mapping Services : Primary Menu

```
SOLVPROD-------------- Mapping Services : Primary Menu -----------------$DD010
 Select Option ===>

    A   - Add Map
    B   - Browse Map
    BS  - Browse Map Source
    U   - Update Map
    D   - Delete Map
    C   - Copy Map
    P   - Print Map
    L   - List Maps
    CM  - Compile Map
    V   - View Map Structure
    X   - Exit

 Map Name ...________________________________ ( Required B BS U D C P V
                                               Optional A CM L )




  F1=Help      F2=Split     F3=Exit      F4=Return
                            F9=Swap
```

The field displayed on the Mapping Services : Primary Menu is as follows:

Map Name

Enter the identifier of a map in support of the relevant menu option (for instance if you select the Update Map option you must specify the map that you want to update.)

A detailed description of the options available on this menu follows.

Adding a Map Definition

Select option **A** from the Mapping Services : Primary Menu to display the Mapping Services : Map Definition panel, as shown in Figure 25-3. Use this panel to define a new map definition.

Figure 25-3. Mapping Services : Map Definition Panel

```
SOLVPROD------------ Mapping Services : Map Definition ----------- Page 1 of 1
 Command ===>                                                  Function=Add

 Map Name ................. ____________________________________

 Description .............. ____________________________________

 Comments ................. ____________________________________________________
                            ____________________________________________________
                            ____________________________________________________
                            ____________________________________________________

 Generated by Appl ID .....
 Source Member Name ....... ________          Number of Source Lines ...
 Source DDNAME ............ ________

 Modification Level .......

 Last Compiled On .........


  F1=Help      F2=Split     F3=File      F4=Save
                            F9=Swap                              F12=Cancel
```

The fields displayed on the Mapping Services : Map Definition panel are as follows:

Map Name

The unique identifier of the map. See Chapter 5, *Naming Standards* for details of how to name a map.

Description

A short description of the map.

Comments

A more detailed description of the map. This is an optional field that can be used to record any notes or additional information about this map.

Source Member Name

The name of the member (from the **Source DD** name**'**s dataset) where the ASN.1 map source is stored.

Source DDNAME

The DD name for the dataset where the ASN.1 map source is stored.

The following display fields are maintained by Mapping Services and cannot be modified:

Generated by Appl ID
The Application ID of the application which generated the map. If an application ID is present, it means that the map is maintained by the application and cannot be modified by the Mapping Services facilities.

Number of Source Lines
The number of lines of source code belonging to this map definition.

Modification Level
A number that is set to 1 when the map is first compiled, and is incremented by one each time the map is recompiled.

Last Compiled On
Contains the date and time when the map was last compiled, and the user ID of the person who performed the compilation.

After completing the Mapping Services : Map Definition panel, press the FILE key to create the map. To cancel the definition, press the CANCEL key.

Maintaining Map Definitions

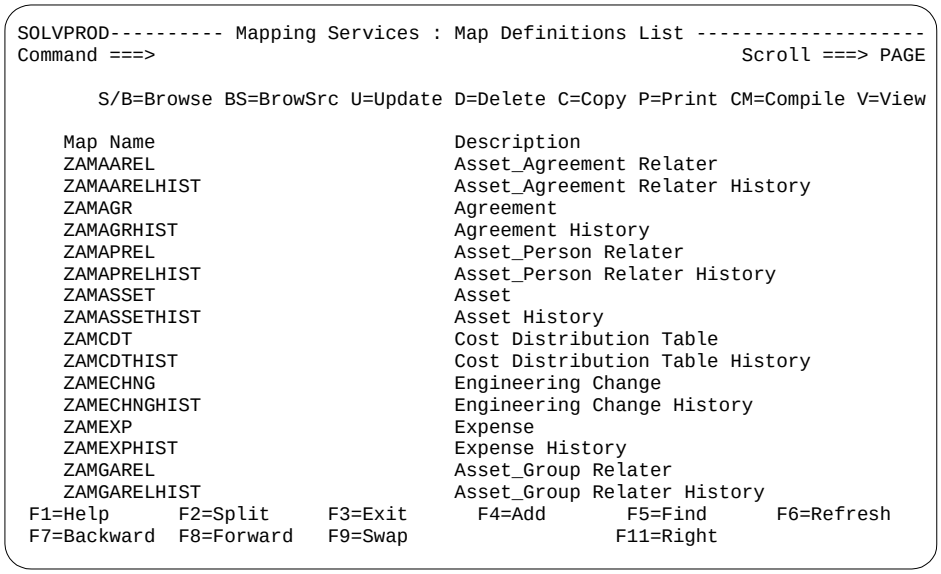
This section describes the maintenance facilities available for existing map definitions.

Listing Map Definitions

Select option **L** to display map definitions that are already defined. If you make an entry (or partial entry) in the **Map Name** field on the menu panel, the list is constrained to map definitions that generically match your entry.

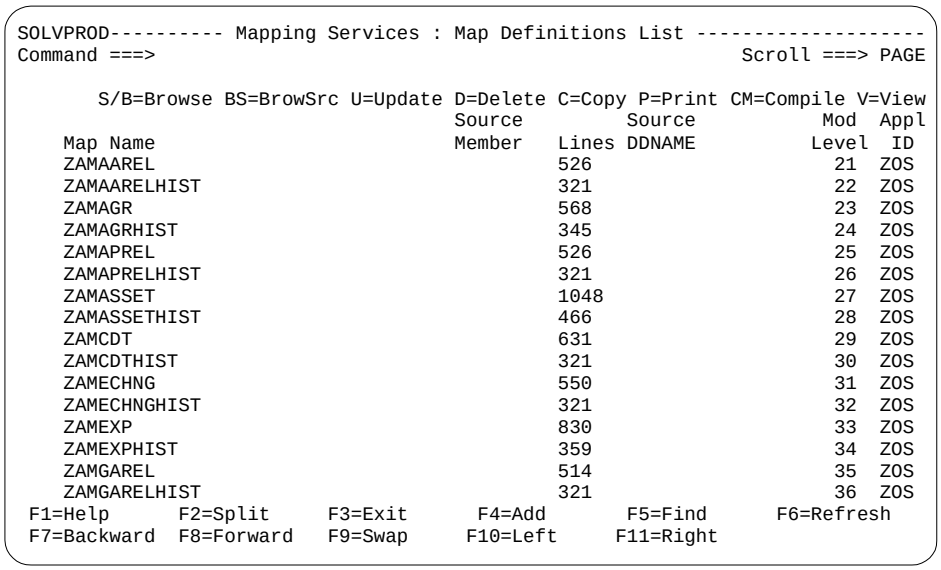
Map definitions are displayed over four panels as shown in Figure 25-4, Figure 25-5, Figure 25-6 and Figure 25-7. Use the RIGHT and LEFT keys to scroll between the panels of the list.

Figure 25-4. Mapping Services : Map Definitions List (page 1)



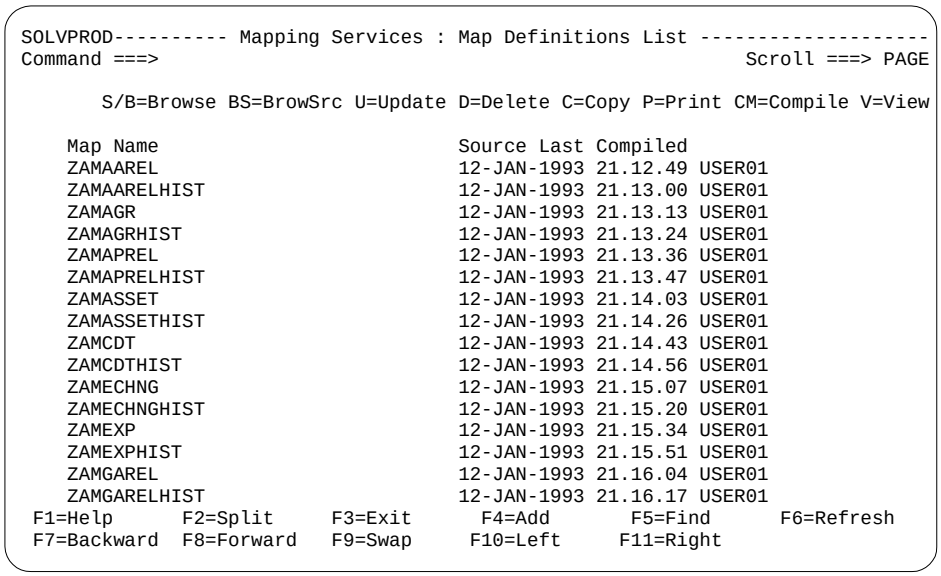
```
SOLVPROD---------- Mapping Services : Map Definitions List --------------------
Command ===>                                                 Scroll ===> PAGE

       S/B=Browse BS=BrowSrc U=Update D=Delete C=Copy P=Print CM=Compile V=View

    Map Name                          Description
    ZAMAAREL                          Asset_Agreement Relater
    ZAMAARELHIST                      Asset_Agreement Relater History
    ZAMAGR                            Agreement
    ZAMAGRHIST                        Agreement History
    ZAMAPREL                          Asset_Person Relater
    ZAMAPRELHIST                      Asset_Person Relater History
    ZAMASSET                          Asset
    ZAMASSETHIST                      Asset History
    ZAMCDT                            Cost Distribution Table
    ZAMCDTHIST                        Cost Distribution Table History
    ZAMECHNG                          Engineering Change
    ZAMECHNGHIST                      Engineering Change History
    ZAMEXP                            Expense
    ZAMEXPHIST                        Expense History
    ZAMGAREL                          Asset_Group Relater
    ZAMGARELHIST                      Asset_Group Relater History
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Figure 25-5. Mapping Services : Map Definitions List (page 2)



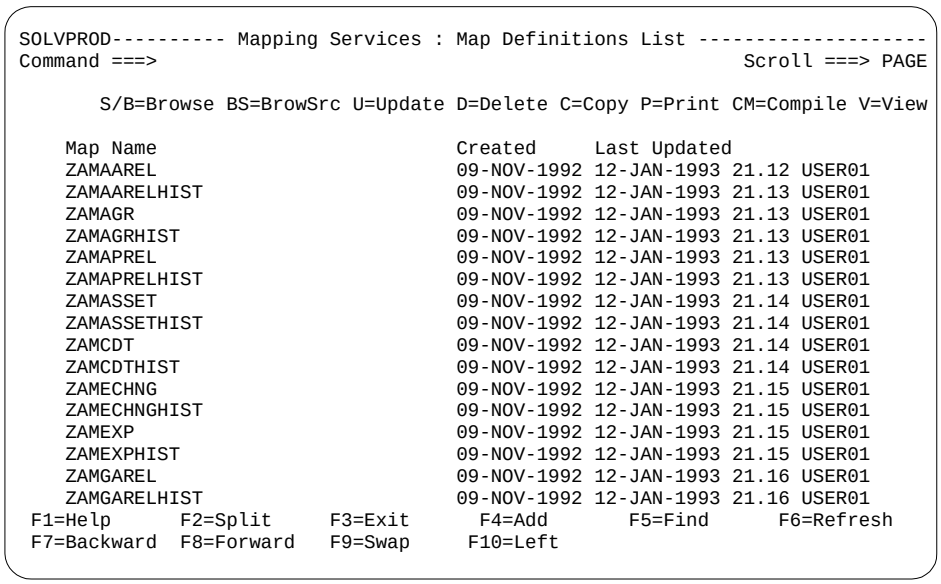
```
SOLVPROD---------- Mapping Services : Map Definitions List --------------------
Command ===>                                                 Scroll ===> PAGE

       S/B=Browse BS=BrowSrc U=Update D=Delete C=Copy P=Print CM=Compile V=View
                                      Source         Source           Mod  Appl
    Map Name                          Member   Lines DDNAME          Level  ID
    ZAMAAREL                                   526                     21  ZOS
    ZAMAARELHIST                               321                     22  ZOS
    ZAMAGR                                     568                     23  ZOS
    ZAMAGRHIST                                 345                     24  ZOS
    ZAMAPREL                                   526                     25  ZOS
    ZAMAPRELHIST                               321                     26  ZOS
    ZAMASSET                                   1048                    27  ZOS
    ZAMASSETHIST                               466                     28  ZOS
    ZAMCDT                                     631                     29  ZOS
    ZAMCDTHIST                                 321                     30  ZOS
    ZAMECHNG                                   550                     31  ZOS
    ZAMECHNGHIST                               321                     32  ZOS
    ZAMEXP                                     830                     33  ZOS
    ZAMEXPHIST                                 359                     34  ZOS
    ZAMGAREL                                   514                     35  ZOS
    ZAMGARELHIST                               321                     36  ZOS
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 25-6. Mapping Services : Map Definitions List (page 3)



```
SOLVPROD---------- Mapping Services : Map Definitions List --------------------
Command ===>                                                 Scroll ===> PAGE

       S/B=Browse BS=BrowSrc U=Update D=Delete C=Copy P=Print CM=Compile V=View

    Map Name                          Source Last Compiled
    ZAMAAREL                          12-JAN-1993 21.12.49 USER01
    ZAMAARELHIST                      12-JAN-1993 21.13.00 USER01
    ZAMAGR                            12-JAN-1993 21.13.13 USER01
    ZAMAGRHIST                        12-JAN-1993 21.13.24 USER01
    ZAMAPREL                          12-JAN-1993 21.13.36 USER01
    ZAMAPRELHIST                      12-JAN-1993 21.13.47 USER01
    ZAMASSET                          12-JAN-1993 21.14.03 USER01
    ZAMASSETHIST                      12-JAN-1993 21.14.26 USER01
    ZAMCDT                            12-JAN-1993 21.14.43 USER01
    ZAMCDTHIST                        12-JAN-1993 21.14.56 USER01
    ZAMECHNG                          12-JAN-1993 21.15.07 USER01
    ZAMECHNGHIST                      12-JAN-1993 21.15.20 USER01
    ZAMEXP                            12-JAN-1993 21.15.34 USER01
    ZAMEXPHIST                        12-JAN-1993 21.15.51 USER01
    ZAMGAREL                          12-JAN-1993 21.16.04 USER01
    ZAMGARELHIST                      12-JAN-1993 21.16.17 USER01
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Figure 25-7. Mapping Services : Map Definitions List (page 4)



```
SOLVPROD---------- Mapping Services : Map Definitions List --------------------
Command ===>                                                 Scroll ===> PAGE

       S/B=Browse BS=BrowSrc U=Update D=Delete C=Copy P=Print CM=Compile V=View

    Map Name                          Created     Last Updated
    ZAMAAREL                          09-NOV-1992 12-JAN-1993 21.12 USER01
    ZAMAARELHIST                      09-NOV-1992 12-JAN-1993 21.13 USER01
    ZAMAGR                            09-NOV-1992 12-JAN-1993 21.13 USER01
    ZAMAGRHIST                        09-NOV-1992 12-JAN-1993 21.13 USER01
    ZAMAPREL                          09-NOV-1992 12-JAN-1993 21.13 USER01
    ZAMAPRELHIST                      09-NOV-1992 12-JAN-1993 21.13 USER01
    ZAMASSET                          09-NOV-1992 12-JAN-1993 21.14 USER01
    ZAMASSETHIST                      09-NOV-1992 12-JAN-1993 21.14 USER01
    ZAMCDT                            09-NOV-1992 12-JAN-1993 21.14 USER01
    ZAMCDTHIST                        09-NOV-1992 12-JAN-1993 21.14 USER01
    ZAMECHNG                          09-NOV-1992 12-JAN-1993 21.15 USER01
    ZAMECHNGHIST                      09-NOV-1992 12-JAN-1993 21.15 USER01
    ZAMEXP                            09-NOV-1992 12-JAN-1993 21.15 USER01
    ZAMEXPHIST                        09-NOV-1992 12-JAN-1993 21.15 USER01
    ZAMGAREL                          09-NOV-1992 12-JAN-1993 21.16 USER01
    ZAMGARELHIST                      09-NOV-1992 12-JAN-1993 21.16 USER01
 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap     F10=Left
```

The fields displayed on the Mapping Services : Map Definitions List are as follows:

Map Name
The identifier of the map.

Description
A short description of the map.

Source Member
The name of the member within the Source DD name's dataset.

Lines
The number of lines of source statements in the map.

Source DDNAME
The identifier of the dataset where the ASN.1 map source is stored.

Mod Level
The modification level of the map definition. This number is incremented by 1 (up to a maximum of 99) each time the map is compiled.

Appl ID
The application identifier of the application which owns the map.

Source Last Compiled
The date and time that the map was last compiled and the identifier of the user who performed the compilation.

Created
The date the map definition was created.

Last Updated
The date and time that the map definition was last updated or compiled and the user ID of the person who performed this.

You can perform the following actions on any item in the list, by placing the appropriate mnemonic next to the item:

Mnemonic	Action
S, B, or /	Browse the selected map definition
BS	Browse the ASN.1 source code for the selected map definition
U	Update the selected map definition
D	Delete the selected map definition
C	Copy the selected map definition
P	Print the selected map definition and its source code
CM	Compile the selected map
V	View the map structure for the selected map definition

You can also perform these functions from the menu by choosing the corresponding menu options. Functions not already discussed are described below.

Browsing a Map Definition

Select option **S**, **B**, or **/** and enter the identifier of an existing map in the **Map Name** field to display the Mapping Services : Map Definition panel in Browse mode.

While in Browse mode, you can press the VIEW key to display the map structure (if the map has been successfully compiled), or the BROWSRC key to display the source code for the map (that is, the source that was saved by the last compilation).

Browsing the ASN.1 Source Code for a Map

Option **BS** displays the ASN.1 source code for the map (from the last successful compile). An example is shown in Figure 25-8.

Figure 25-8. Browsing ASN.1 Source Code for a Map.

```
SOLVPROD----------- Mapping Services : ASN1 Map File Source ----Columns 001 072
Command ===>                                  Function=Browse Scroll ===> PAGE

Map Name ...... $OSSDU                          Source Member ... $OSSDU
Description ... Service Data Unit               Source DDNAME ... COMMANDS

LINE ----+----10---+----20---+----30---+----40---+----50---+----60---+----70---
**** ***************************** TOP OF DATA ******************************
0001 --     ++INCLUDE #OSVERSD
0002 --*************************************************************************
0003 --                                                                        *
0004 -- NAME        : $OSSDU                                                  *
0005 --                                                                        *
0006 -- DESCRIPTION : ASN.1 map for Service Data Unit used in calling         *
0007 --               Object Services                                          *
0008 --                                                                        *
0009 -- CREATE DATE : 31-OCT-1991                                             *
0010 --                                                                        *
0011 --*************************************************************************
0012
0013 --     ++INCLUDE #NMCOPYD
0014
 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Updating a Map Definition

Select option **U** and specify the map definition in the **Map Name** field to display the Mapping Services : Map Definition panel in Update mode. Update the map definition as required; refer to the section titled *Adding a Map Definition*, on page 25-5 for details.

While in Update mode, you can press the VIEW key to view the map structure, or BROWSRC to display the source code for the map (from the last successful compile).

Deleting a Map Definition

Select option **D** and specify the map definition in the **Map Name** field to delete the map definition.

After doing the above,the map definition is displayed with a message requiring you to confirm the deletion. Press ENTER to delete the map definition, or press the CANCEL key to cancel the deletion.

Copying a Map Definition

To copy an existing map definition to another (new) map definition select option **C** from the Mapping Services : Primary Menu, and provide the **Map Name** of the map definition to copy from.

The Mapping Services : Map Definition panel is displayed as shown in Figure 25-3; update the new map definition as required (refer to the section titled *Adding a Map Definition* for details).

After completing the new map definition, press the FILE key. To cancel the copy, press the CANCEL key.

Printing a Map Definition and its ASN.1 Source Code

Option **P** displays the PSM : Confirm Printer panel for you to enter the required printing details. Press the CONFIRM key or press the CANCEL key.

Compiling a Map's ASN.1 Source Code

Select option **CM** from the Mapping Services : Primary Menu to compile the map.

The map definition is displayed when you do this. Check (and modify if necessary) the **Source Member** and **DDName** fields. You can use the BROWMEM and BROWSRC keys to browse the member definition and source definition, respectively, from this panel.

Press the ACTION key to compile the map.

If any warnings or errors occur, the Mapping Services : Compiler Messages screen is displayed as shown in Figure 25-9.

Figure 25-9. Mapping Services : Compiler Messages.

```
SOLVPROD------------ Mapping Services : Compiler Messages ---------------------
 Command ===>                                                 Scroll ===> PAGE

                                                                        S/H=Help
    Error and/or Warning Messages for ASN.1 Compile
    DD0904 THE FOLLOWING MESSAGES WERE ISSUED BY COMPILER FOR MAP $OSSDU
    DD0361 Warning: line 147 - unnamed component
    **END**









  F1=Help      F2=Split     F3=Exit                  F5=Find      F6=Refresh
  F7=Backward  F8=Forward   F9=Swap                 F11=Right
```

Viewing a Map Structure

Select option **V** to display the compiled map structure of the map definition that you specify in the **Map Name** field. An example is shown in Figure 25-10.

Figure 25-10. Mapping Services : View Map Structure Panel.

```
SOLVPROD------------ Mapping Services : View Map Structure --Line 1 to 16 of 85
Command ===>                                  Function=Browse Scroll ===> CSR

Map Name ...... ZAMAAREL

******************************** TOP OF DATA ********************************
CLASSID
NAMEID
NAME.ZOSSEQID
NAME.ZOSSEQPOS
NAME.ZMGRELID
ATTR.ZAMAGR
ATTR.ZAMASSET
ATTR.ZAMRAGDESC
ATTR.ZAMRAGTYPE
ATTR.ZAMRAG1GRP
ATTR.ZAMRAG2GRP
ATTR.ZAMRASPROD
ATTR.ZAMRASREF
ATTR.ZMGRELID
ATTR.ZOSCLASS
ATTR.ZOSCOM
 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right
```

Part VI

Reference Material

A

Customer Support Services

Sterling Software provides a variety of customer support services to ensure that you get the most out of your Sterling Software products. The sections that follow provide you with information to make it easier for you to communicate with our Customer Services staff. The sections include our telephone hotline and fax numbers, Support Center addresses, and outline basic information you should have on hand when calling your local Support Center for assistance. If you have questions other than product support, our Technical Support Representatives can also help you get the information you need.

If you have questions about billing, new product orders, or the Product Support and Enhancement Agreement, contact your local Sterling Software representative.

Technical Support

If you encounter a problem with one of our products that you cannot resolve by reading and following the online help or the documentation, call your local Sterling Software Support Center. Our Technical Support Representatives are available to answer your questions about Sterling Software systems.

To receive prompt service, use the following hotline numbers for the countries and products listed. If you are outside these countries, contact your local Sterling Software representative for help.

In most regions, you can call the hotline number outside of prime support hours if your problem is critical and cannot wait until the next business day. We have a Technical Support Representative on call 24 hours a day, 365 days a year for such emergencies.

If the Technical Support Representative asks you to send information or documentation regarding your problem to the Support Center, use your local Support Center's fax number or mailing address.

You can help the Technical Support Representative solve your problem more quickly if you have the following information at hand before you call:

- Your system model, make, and serial number
- Your problem tracking number and customer number, as these apply
- Your operating system, release level, and maintenance level
- Release and maintenance level of the product or component causing the problem
- Final error messages, especially the message numbers, that the problem caused
- All product manuals for easy reference

If all Technical Support Representatives are busy and you have to leave a message, remember to provide your name, your company name, phone number, call incident or problem number if you have one, and a brief description of the problem. You can also leave other relevant information such as how to reach you if you are not available at the stated phone number. A representative will return your call as soon as possible.

Sterling Software Web Site

The Network Management Division of Sterling Software, Inc. maintains an extensive site on the World Wide Web at *http://www.solve.sterling.com*.

The site features a Support page that lists many customer resources, including an Online Services facility for all SOLVE customers enrolled in the Software Support Program. You need to obtain a special user ID and password from your Customer Service Representative in order to access the Online Services facility.

Local Support Centers

Sterling Software Support Centers are listed alphabetically by region below. When calling for technical assistance, please use the Support Center in your region. If you are outside these countries, contact your local Sterling Software representative for help.

Europe

Denmark

Direct Support Hotline:	+ 45 44 209 951
Prime Support Hours:	Normal business hours Monday through Friday
Fax:	+ 45 44 209 950
Mailing Address:	Sterling Software Lautruphoj 1 - 3 2750 Ballerup Danmark

France

Direct Support Hotline:	+ 33 1 47 96 74 73
Prime Support Hours:	Normal business hours Monday through Friday
Fax:	+ 33 1 47 96 74 75
Mailing Address:	Paris Customer Support Center Sterling Software Tour Framatome La Defence 6 92084 - Paris La Defence

Germany

Direct Support Hotline:	+ 49 6102 709 0
Support Hours:	Normal business hours Monday through Friday
Fax:	+ 49 6102 709111
Mailing Address:	Technischer Kundenservice (TKS) Sterling Software GmbH Martin-Behaim-Strasse 12 D-63263 Neu-Isenburg

Italy

Direct Support Hotline:	+ 39 11 771 4095
Support Hours:	Normal business hours Monday through Friday
Fax:	+ 39 11 743 878
Mailing Address:	Sterling Software Srl Corso Svizzera 185 10149 Torino

Norway

Direct Support Hotline:	+ 47 22 65 10 52
Prime Support Hours:	Normal business hours Monday through Friday
Fax:	+ 47 22 64 66 79
Mailing Address:	Sterling Software Brynsveien 13 Postboks 6392 Etterstad N-0604 Oslo

Spain

Direct Support Hotline:	+ 34 1 598 35 05
Prime Support Hours:	Normal business hours Monday through Friday
Fax:	+ 34 1 556 92 12
Mailing Address:	Edificio Alfredo Mahou, planta 7? B Plaza Manuel Gomez Moreno 2 28020 Madrid - Spain

United Kingdom

Direct Support Hotline:	+ 44 1932-587 575
Prime Support Hours:	Normal business hours Monday through Friday
Fax:	+ 44 1932-587 576
Mailing Address:	Chertsey Customer Support Center Sterling Software Sterling Court Eastworth Road Chertsey Surrey KT16 8DF

North America

Canada
Contact US Support Center

United States

Direct Support Hotline:	
Metro Washington	+ 1 703 264 8334
Continental US	+ 1 800 663 6529
Prime Support Hours:	8:30 a.m. to 5:30 p.m. EST Monday through Friday
Fax:	+ 1 703 264 0751
Mailing Address:	Reston Customer Support Center Network Management Division Sterling Software 1800 Alexander Bell Drive Reston, VA 20191

South America

Brazil

Direct Support Hotline:	+ 55 11 5505 3366
Prime Support Hours:	08:30am through 17:30pm Monday through Friday
Fax:	+ 55 11 5505 1805
Mailing Address:	Sterling Software Av. das Nacoes Unidas, 12.995 - 22 andar - Brooklyn Novo 04578-000 Sao Paulo SP Brasil

Pacific Rim

Australia or New Zealand

Direct Support Hotline:	+ 61 2 9975 8469
Prime Support Hours:	8:30 a.m. to 5:00 p.m. EST Monday through Friday
Fax:	+ 61 2 9975 5245
Mailing Address:	Sydney Customer Support Center Sterling Software Forest Corporate Park 28 Rodborough Road Frenchs Forest NSW 2086

Japan

Direct Support Hotline:	+ 81 3 5802 9112
Prime Support Hours:	9:00 a.m. to 5:30 p.m. Monday through Friday
Fax:	+ 81 3 5802 9100
Mailing Address:	Sterling Software Sumitomo Fudosan Korakuen Building 1-4-1 Koishikawa Bunkyou-ku, Tokyo 112-0022

Singapore

Direct Support Hotline:	+ 65 736 1162
Prime Support Hours:	Normal business hours Monday through Friday
Fax:	+ 65 325 1163
Mailing Address:	Sterling Software 133 Cecil Street #09-02 Keck Seng Tower Singapore 069536

User Conferences

Sterling Software holds annual user conferences at locations throughout North America. These conferences give you an opportunity to learn from the experiences and expertise of other Sterling Software customers and to share some of your product knowledge with them. Each conference includes technical sessions, user presentations, roundtable discussions, and time for informal exchanges between customers and Sterling Software employees.

For more information about our user conferences, contact your local Sterling Software representative.

B

Text Editor Commands

This chapter describes the Text Editor commands.

Using the Text Editor Commands

On the left of each line of text is a *sequence number* field. When updating or adding text, *line commands* are entered into these fields to perform edit functions. For a comprehensive list of text editor line commands and a description of the use of each, see the section titled *Line Commands.*

Primary commands are entered into the **Command** field; see the section titled *Primary Commands* for details.

Line Commands

The line on which the line command is entered is referred to as the current line in the following text.

The following line commands are supported:

A*nn*

Insert copied or moved line(s) after the current line, *nn* times. If *nn* is omitted, the line(s) will be inserted once only.

B*nn*

Insert copied or moved line(s) before the current line, *nn* times. If *nn* is omitted, the line(s) will be inserted once only.

C*nn*

Copy *nn* lines starting from the current line. Enter A or B beside the line of text to which the lines are to be copied after or before, or O*nn* or OO to copy the text over a sequence of lines.

If *nn* is omitted, only the current line is copied.

CC

Copy a sequence of lines. Enter CC beside the first and last line to be copied and A or B beside the line of text to which the lines are to be copied after or before, or O*nn* or OO to copy the text over a sequence of lines.

COLS

A line containing column numbers is inserted after the current line.

D*nn*

Delete *nn* lines starting from the current line. If *nn* is omitted, only the current line is deleted.

DD

Delete a sequence of lines. Enter DD beside the first and last line to be deleted.

I*nn*

Insert *nn* blank lines after the current line. If *nn* is omitted, one blank line is inserted.

ID

Inserts lines of text containing the current date and time, the user's ID, name and phone number.

LC*nn*

Convert alphabetic characters to lowercase on the next *nn* lines. If *nn* is omitted, only the current line is converted.

LCLC

Convert alphabetic characters to lowercase in a block of lines. Enter LCLC beside the first and last lines to convert.

M*nn*

Move *nn* lines starting from the current line. Enter A or B beside the line of text to which the lines are to be moved after or before, or O*nn* or OO to move the text over a sequence of lines.

If *nn* is omitted, only the current line is moved.

MM

Move a sequence of lines. Enter MM beside the first and last line to be copied and A or B beside the line of text to which the lines are to be moved after or before, or O*nn*, to move the text over a sequence of lines.

NA

Insert the contents of the Notepad after the current line.

NB

Insert the contents of the Notepad before the current line.

N*nn*

Append *nn* lines to the Notepad. If *nn* is omitted, only the current line is appended to the Notepad.

NN

Append a sequence of lines to the Notepad. Enter NN beside the first and last line to be appended to the Notepad.

O*nn*

Overlay copied or moved line(s) over the next *nn* lines. If *nn* is omitted, it defaults to the number of lines being moved or copied.

OO

Overlay copied or moved line(s) over a sequence of lines. Enter OO beside the first and last lines to be overlaid.

QA

Insert queued lines after the current line.

QB

Insert queued lines before the current line.

Q*nn*

Queue *nn* lines. If *nn* is omitted, only the current line is queued. This command overwrites currently queued lines.

Q+*nn*

Append *nn* lines to the end of the queue. If *nn* is omitted, only the current line is appended.

QQ

Queue a sequence of lines. Enter QQ beside the first and last line to be queued. This command overwrites currently queued lines.

QQ+

Append a block of lines to the end of the queue. Enter QQ+ beside the first and last lines to be appended to the queue.

R*nn*

Repeat the current line *nn* times. If *nn* is omitted, the current line is repeated once.

RR*nn*

Repeat a sequence of lines *nn* times. Enter RR*nn* beside the first and last line to be repeated. If *nn* is omitted, the sequence of lines is repeated once only.

TE

Text entry mode. Blank input lines are inserted after the current line allowing the entry of many lines of text. When the ENTER key is pressed the text is flowed within the current margins and inserted.

TF

Flow text to the current right margin preserving the indentation of each line from column 1. Text is flowed from the current line to the end of the paragraph. The end of the paragraph is determined by a blank line or a change in indentation.

TF*nn*

This command is the same as TF except that it uses the right margin specified by *nn*.

TFM

Flow text to the current left and right margins preserving only the relative indentation between the current line and the following line. Text is flowed from the current line to the end of the paragraph. The end of the paragraph is determined by a blank line or a change in indentation.

TM

Merge text. The current line is split at the cursor position as for a TS command. Text Entry mode is then invoked. On exit from Text Entry mode a TF is performed from the current line through to the end of the paragraph.

TS

Split text. A new line is inserted after the current line and the text from the cursor position to the end of the line is moved to the new line.

UC*nn*

Convert alphabetic characters to uppercase on the next *nn* lines. If *nn* is omitted, only the current line is converted.

UCUC

Convert alphabetic characters to uppercase in a block of lines. Enter UCUC beside the first and last lines to convert.

)*nn*

Shift the data on the current line *nn* positions to the right. Any data shifted beyond the maximum record length is lost. If *nn* is omitted, the data is shifted one position to the right.

(*nn*

Shift the data on the current line *nn* positions to the left. Any data shifted beyond column one is lost. If *nn* is omitted, the data is shifted one position to the left.

))*nn*

Shift the data on a block of lines *nn* positions to the right. Enter *))nn* beside the first and last lines to have their data shifted. Any data shifted beyond the maximum record length is lost. If *nn* is omitted, the data is shifted one position to the right.

((*nn*

Shift the data on block of lines *nn* positions to the left. Enter *((nn* beside the first and last lines to have their data shifted. Any data shifted beyond column one is lost. If *nn* is omitted, the data is shifted one position to the left.

Figure B-1, Figure B-2, and Figure B-3 show some examples of commonly used line commands.

Figure B-1. Adding Lines in the Text Editor - part 1

```
SOLVPROD-------------------- CAS : Notepad Editor -------------Line 1 to 1 of 1
Command ===>                                      Function=Update Scroll ===> PAGE

**** ****************************** TOP OF DATA ******************************
i501 To add lines, type 'i' plus the number of lines to add in left column
**** ***************************** BOTTOM OF DATA *****************************









 F1=Help      F2=Split     F3=File      F4=Save      F5=Find      F6=Change
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure B-2. Adding Lines in the Text Editor - part 2

```
SOLVPROD-------------------- CAS : Notepad Editor -------------Line 1 to 6 of 6
Command ===>                                    Function=Update Scroll ===> PAGE

**** ****************************** TOP OF DATA ******************************
0001 To add lines, type 'i' plus the number of lines to add in left column
0002
0003 In this example 'i5' was typed into the left column, adding 5 lines to
0004 the Notepad text entry area.
0005
0006
**** ***************************** BOTTOM OF DATA ******************************









 F1=Help      F2=Split     F3=File      F4=Save      F5=Find      F6=Change
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure B-3. Other Line Commands in the Text Editor

```
SOLVPROD-------------------- CAS : Notepad Editor -----------Line 1 to 14 of 14
Command ===>                                    Function=Update Scroll ===> PAGE

**** ****************************** TOP OF DATA ******************************
0001
d002 A letter 'd' in the first column will delete that one line
0003
dd04 To delete a block of text type 'dd' into the left column of the first line
0005 of the block to be deleted.....
dd06 .....then type 'dd' in the last line of the block and press ENTER.
0007
c008 To copy one line of text type 'c' into the left column.
0009
cc10 To copy a block of text type 'cc' in the first line of the block...
cc11 .....then type cc in the last line of the block and press ENTER.
0012
a013 The letter 'a' in the left column will insert copied text after 'this line'
0014
b015 The letter 'b' will insert copied text before 'this line'
**** ***************************** BOTTOM OF DATA ******************************


 F1=Help      F2=Split     F3=File      F4=Save      F5=Find      F6=Change
 F7=Backward  F8=Forward   F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Primary Commands

In addition to the standard SOLVE commands, the following primary commands are available to you within the text editor:

ALL *string*

Positions on the first occurrence of the specified string and displays a message indicating the total number of occurrences that are found.

CHANGE ***from-string to-string*** **[ALL]**

Modifies a string of characters in the text. Enter CHANGE (or C) followed by the character string to be changed, followed by the new character string, followed by ALL, if every occurrence of character string is to be changed. If either character string contains imbedded blanks, enclose the character string in single or double quotes. If either character string contains quotes, the entire string must be enclosed by the quote which does not appear in the string. To repeat the change enter the CHANGE command with no operands. Change processing starts from the current cursor position.

Note

If the from-string contains embedded blanks, the string is not changed unless it is contained entirely within a single line.

FIND ***string***

Searches for a string of characters in the text. Enter FIND (or F) followed by the character string. To repeat the search press the FIND function key or enter the FIND command with no operands. Searching starts from the current cursor position.

Note

If the string being searched for contains embedded blanks, it must be enclosed in single or double quotes. In addition, the string is not found unless it is contained entirely within a single line.

FIRST ***string***

Starts a search at the top of the text and finds the first occurrence of the specified string.

FLOW

Flows all text to the right margin preserving the indentation of each line from column 1. To flow text to a specific right margin enter FLOW, followed by the right margin column number. To flow text to the current right margin enter FLOW with no operands.

FLOWM

Flows all text to the left and right margins preserving only the relative indentation between the first and second lines of each paragraph. To flow text within specific margins enter FLOWM, followed by the left margin and right margin column numbers separated by a space.

LAST ***string***

Starts a search at the bottom of the text and finds the first occurrence of the specified string.

LCMD ***cmd***

Executes the line command *cmd*. This can be used to assign often used line commands to function keys.

LM *nn*

Sets the left margin of the text. Enter LM followed by the required left margin column number.

MARGINS

Displays/sets the left and right margins of the text. To set the margins enter MARGINS or MAR, followed by the left margin column number, followed by the right margin column number. To display the current margin settings enter MARGINS or MAR with no operands. Margin settings are only used when flowing text as a result of the FLOW or FLOWM primary command or the TF, TFM or TE line commands. The difference between the left and right margins must not be less than 20.

NEXT *string*

Searches forward through the text to find the next occurrence of the specified string. This is the default.

NULLS

Pads text lines with nulls or blanks. To pad text lines with blanks enter NULLS OFF. To pad text lines with nulls enter NULLS ON or NULLS.

NOTEPAD

Gives access to the Notepad facility which holds temporary text entered during the session.

PREV *string*

Searches backward through the text to find the previous occurrence of the specified string. This is the default.

PRINT

Prints the text on the printer of your choice.

RESET

Clears all pending line commands. This command can be abbreviated to RES.

RM *nn*

Sets the right margin of the text. Enter RM followed by the required right margin column number.

C

Panel Editor

This appendix describes the *Primary Commands* and *Line Commands* available for editing a panel definition.

- Primary commands are entered on the command line.
- Line commands are single-character editing commands which are entered in a column on the left of the screen to action lines of text.

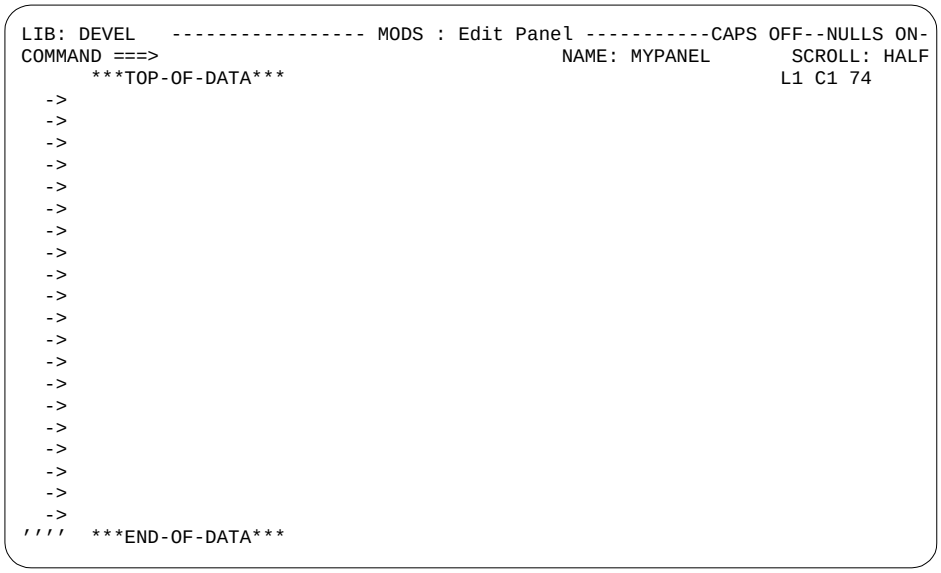
Adding a Panel Definition

Figure C-1 shows the MODS : Edit Panel when adding a new panel called MYPANEL. The definition is specified using a series of lines of text which contain panel commands.

Each line of text is preceded by the arrow (->) characters, within the `***TOP-OF-DATA***` and `***END-OF-DATA***` delimiters.

Once data from insert lines is entered, the arrow is converted to ’’’’ to show the data is accepted. If data is not entered into a line, that line is ignored when the data is saved. Additional lines can be inserted using the I (Insert) line command.

Figure C-1. MODS : Edit Panel



The fields displayed on the MODS : Edit Panel are as follows:

LIB:*xxxxxx*
The name of the library containing the member being edited.

MODS : Edit Panel
The title for the panel edit screen.

CAPS OFF/ON
Indicates whether or not data must be translated into upper case characters.

NULLS ON/OFF
Indicates whether lines are padded with blanks or nulls.

COMMAND
The *primary command* field.

NAME: *xxxxxxx*
The name of the panel being edited.

SCROLL: HALF/PAGE
The current scrolling amount.

L*n*
The line number of the first line of panel data currently displayed.

Cn
The column number of the leftmost position of panel data currently displayed.

nn

The column number of the rightmost position of panel data currently displayed.

The screen also includes a message area along the top of the screen, where error and informational messages are displayed.

Terminating a Panel Edit

Terminate an edit session by pressing the EXIT or RETURN key. This saves the data if any modifications are made. Before data is saved, any outstanding line commands and primary commands are processed, any remaining insert lines which do not contain data are removed, and any column ruler lines inserted by the COLS line command are removed.

The SAVE command can be used to save data without terminating your edit.

If data is wrongly modified, use the CANCEL command to exit the panel and cancel all changes made since the beginning of the edit session, or since the last SAVE command.

Scrolling Edit Data

You can scroll the data being edited in four directions—UP, DOWN, LEFT, and RIGHT. The SCROLL value in the top right corner of the screen determines the amount of movement performed for a scroll, and can be set as follows:

Table C-1. Scroll Amounts

Mnemonic	Amount
PAGE	Scroll a full screen
HALF	Scroll half a screen
MAX	Scroll to the beginning or end of the data
nnn	Scroll *nnn* rows or columns (where *nnn* is a number between 1 and 999)

The **Scroll** field defaults to HALF when an edit session starts. To set a new scroll value, enter the required value in the **Scroll** field. For HALF or PAGE, you need only enter H or P.

To make a once-only change to the scroll amount, enter the required scrolling amount in the **Primary Command** field and press the required function key. This does not affect the value in the **Scroll** field.

Format of Edit Panel

Panel Edit supports a maximum window depth of 43 lines (as for 3270 Model 4 devices). If the operational window is smaller than 43 lines, the edit panel is adjusted to operate within that size.

Homing the Cursor (F12/F24)

Homing is the term used to reposition the cursor at the **Primary Command** field, using the F12 or F24 function keys.

When a homing function occurs, no other command processing is performed.

Panel Editor Primary Commands

Primary commands are entered in the **Primary Command** field. Primary commands enable you to do the following:

- Copy or merge other members with this panel definition
- Create or replace another member
- Reset line commands
- Cancel edit changes
- Scroll through screen data—up, down, left, right
- Find a specified character string
- Change a specified character string
- Save data
- Control the edit environment

Primary commands can be abbreviated to their minimum unique string. For example, the SAVE command can be abbreviated to S as there is no other primary command commencing with S. The REPLACE command however, can only be abbreviated to REP, because a shorter abbreviation would conflict with the RESET command.

Primary commands can be entered in either upper or lower case—they are translated to upper case before use. Exceptions to this are the FIND and CHANGE commands, which are described below.

Table C-2. Panel Editor Primary Command Summary

Command	Description
BOTTOM	Move to bottom of data
CANCEL	Terminate edit without saving data
CAPS	Set upper/lower case translation option
CHANGE	Change the specified string
COPY	Copy another panel
CREATE	Create a new panel
DOWN	Scroll towards the bottom of the data
FIND	Locate the specified string
FSM	Display panel in Full Screen Mode
HALF	Temporarily override the scroll amount
LEFT	Scroll towards the left of the data
MAX	Temporarily override the scroll amount
MOVE	Copy another panel, then delete the original
NULLS	Set nulls option
PAGE	Temporarily override the scroll amount
QFREE	Delete saved block
REPLACE	Replace an existing panel
RESET	Reset line commands and insert lines
RIGHT	Scroll towards the right of the data
SAVE	Save the panel
TOP	Move to the top of the data
UP	Scroll towards the top of the data

BOTTOM

Scrolls to the last page of data, where the `***END-OF-DATA***` indicator is displayed as the last line in the display.

CANCEL

Terminates the current edit session and discards any changes to data since the last SAVE command, or since the beginning of the edit session. Any pending or incomplete line commands are ignored.

CAPS [ON | OFF]

Determines if alphabetic data is automatically translated into upper case. **ON** is assumed, if no operand is entered.

If DBCS support is specified (by setting SYSPARMS DBCS), the CAPS primary command is treated as invalid—the CAPS status is always CAPS OFF.

CAPS mode also determines whether character strings entered as an operand for the CHANGE or FIND commands are translated to uppercase. If CAPS OFF is in effect, character strings are not translated to upper case, and the CHANGE and FIND commands search for the string exactly as entered.

When data is saved the current CAPS setting is remembered and reinstated when the member is next edited. When an edit session for a new member starts, the default setting for CAPS is determined by the setting of the EDITCAPS operand of the SYSPARMS command.

CHANGE ***from-string to-string*** [*left-col right-col*] [ALL]

Scans for occurrences of the specified *from-string* and changes it to the *to-string***.** Scanning starts from the current cursor location and proceeds until the string is found or the bottom of the data is reached. The strings are translated to upper case if operating in CAPS ON mode (see the CAPS command).

If the strings contain blanks or quotes, they must be entered in either single or double quotes. For example:

```
CHANGE ‘my data’ ‘HIS DATA’

CHANGE ”it’s” ”it isn’t”
```

ALL specifies that all occurrences of the *from-strin**g*** are to be changed. *Left-col* and *right-col* establish left and right column boundaries that delimit the change. They must be entered as numeric values in the range 1 to 132. If no boundaries are set, all columns are searched. For example:

```
CHANGE XYZ ABC 30 40 ALL
```

Specifies that all occurrences of the *from-string* XYZ is changed to the *to-string* ABC between columns 30 and 40.

The CHANGE command can be used to insert entire strings of data by specifying the *from-string* as a null value. For example:

```
CHANGE ‘’ ‘THIS IS A NEW COLUMN’ 30 ALL
```

Indicates that a new column of data is to be inserted in column 30. Existing data to the right of column 30 is shifted right and truncated if necessary. (If no *left-col* is specified, column 1 is assumed.)

Similarly, the CHANGE command can be used to remove a block of data by changing the *from-string* to a *null string*. For example:

```
CHANGE 'OLD DATA' '' 40 ALL
```

Data to the right of the *from-string* is moved to the left.

F6 can be used to repeat the CHANGE command. You can use the repeat FIND function key (F5) to find the next occurrence of the string before using F6 to actually make the change.

COPY *membername*

Copies another panel (*membername*) into the current panel. The COPY command works in conjunction with either an A (after), or B (before) line command, which designates the point where the copied panel is to be inserted.

Note

The COPY command only copies a panel which is a member of the library currently being edited.

Figure C-2 shows an example of a COPY command that copies a member called PANEL2 after the second data line, when the ENTER key is pressed.

Figure C-2. COPY Command Example

```
LIB: DEVEL   ----------------- MODS : Edit Panel -----------CAPS OFF--NULLS ON-
COMMAND ===> COPY PANEL2                       NAME: PANEL1       SCROLL: HALF
      ***TOP-OF-DATA***                                           L1 C1 74
'''' This is line 1 of my data
A''' This is line 2 of my data
'''' This is line 3 of my data
'''' This is line 4 of my data
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
  ->
''''  ***END-OF-DATA***
```

CREATE *membername*

Creates a new panel (*membername*), in the current library, by moving or copying specified lines from the panel you are editing.

The CREATE command works in conjunction with either the C (Copy), or M (Move) line command. C specifies that lines are to be copied and left in their current location. M specifies that after the lines are copied they are to be deleted from their current position.

To copy all lines to the end of the panel, you can enter C999 (without needing to know the precise number of lines).

In Figure C-3, a CREATE command moves 10 lines (starting from the second line of data) and creates a member called PANEL3 in the current library. Figure C-4 shows the data after the CREATE is completed.

Figure C-3. CREATE Command Example

```
LIB: DEVEL   ----------------- MODS : Edit Panel -----------CAPS OFF--NULLS ON-
COMMAND ===> CREATE PANEL3                    NAME: PANEL1       SCROLL: HALF
      ***TOP-OF-DATA***                                          L1 C1 74
'''' This is line 1 of my data
M10' This is line 2 of my data
'''' This is line 3 of my data
'''' This is line 4 of my data
'''' This is line 5 of my data
'''' This is line 6 of my data
'''' This is line 7 of my data
'''' This is line 8 of my data
'''' This is line 9 of my data
'''' This is line 10 of my data
'''' This is line 11 of my data
'''' This is line 12 of my data
'''' This is line 13 of my data
'''' This is line 14 of my data
  ->
  ->
  ->
  ->
  ->
  ->
  ->
```

Figure C-4.   Data After CREATE Command Has Deleted 10 Lines

```
LIB: DEVEL   ----------------- MODS : Edit Panel -----------CAPS OFF--NULLS ON-
 COMMAND ===>                                  NAME: PANEL1       SCROLL: HALF
 N56305 MEMBER PANEL3 CREATED                                     L1 C1 74
 '''' This is line 1 of my data
 '''' This is line 12 of my data
 '''' This is line 13 of my data
 '''' This is line 14 of my data
   ->
   ->
   ->
   ->
   ->
   ->
   ->
   ->
   ->
   ->
   ->
   ->
   ->
   ->
 ''''  ***END-OF-DATA***
```

DOWN

Scrolls towards the end of the data by the amount specified in the **Scroll** field. The DOWN command is assigned to F8 (and F20).

FIND *string*

Scans for the occurrence of a specific character string. Scanning starts from the current cursor position and proceeds until the string is found or the bottom of the data is reached. You can use F5 to find the next occurrence of the string.

The string is translated to upper case if operating in CAPS ON mode (see the CAPS command). If the string to be found contains blanks, it must be entered in quotes (single or double). For example:

```
FIND 'my data'
```

FSM

Dedicates the entire screen for use as a data entry area. FSM mode is invoked by typing FSM in the command line and positioning the cursor on the line that is to become the top line of the display. Access to column 80 onwards is gained by using the RIGHT scroll key. If the panel being created is longer than the screen, you can access hidden rows by using the UP and DOWN function keys. You can return to normal mode using F3.

HALF

Scrolls the edit panel one half page in the direction specified. The HALF command is used with one of the scrolling keys (F7— UP, F8—DOWN, F10—LEFT, and F11—RIGHT).

The number of lines scrolled depends on the window in which the editing is being performed. If 20 data lines are displayed, the HALF command scrolls 10 lines.

LEFT

Scrolls the edit panel to the left hand side of the displayed data by the amount specified in the **Scroll** field. The LEFT command is assigned to F10.

MAX

Scrolls the maximum amount in the direction specified. The MAX command is used with one of the scrolling keys (F7—UP, F8—DOWN, F10—LEFT, and F11—RIGHT).

MOVE *membername*

Moves another panel (*membername*) into the current panel. The MOVE command works in conjunction with either an A (after) or B (before) line command, which designates the point where the copied data is to be inserted.

The MOVE command operates in the same way as the COPY command with the additional function of deleting the panel from the library once it has been copied (Figure C-2 is an example of a COPY command).

Note

The MOVE command only moves a panel which is a member of the library currently being edited.

NULLS [ON | OFF]

Turns nulls on or off. (The current NULLS status can be determined from the information area at the top of the Edit panel.)

When data is saved, the current NULLS setting is remembered and reinstated when the panel is next edited. When you edit a new panel, the default setting for NULLS is determined by the setting of the SYSPARMS command EDITNULL operand.

Note

If NULLS ON is in effect, and the cursor is positioned beyond the end of data in a line, then, when additional characters are added, this new data is shifted left by the terminal hardware to join up with the existing data to the left. You can use the space bar to insert blanks in place of the nulls to the right of existing data in the line.

Alternatively, the cursor can be positioned at the required location (past the last data character on the line) and the ENTER key pressed. Blanks are added to the line up to the cursor location without you having to use the space bar.

PAGE

Scrolls one page in the direction specified. The PAGE command is used with one of the scrolling keys (F7—UP, F8—DOWN, F10—LEFT, and F11—RIGHT).

The amount scrolled is determined by the size of the window where the editing is being performed. If 15 data lines are displayed, the PAGE command scrolls 15 lines.

QFREE

Removes text saved by an earlier Q line command from the temporary storage area. See the Q line command description for details.

REPLACE *membername*

Replaces another panel (*membername*), in the current library, with the specified lines from the panel you are editing.

The REPLACE command works in conjunction with either the C (Copy) or M (Move) line command. C specifies that lines are to be copied and left in their current location. M specifies that after the lines are copied they are to be deleted from their current position.

To copy all lines to the end of the panel, you can enter C999 (without needing to know the precise number of lines).

The REPLACE command works in the same way as the CREATE command, except that it overwrites an existing panel instead of creating a new one. See Figure C-3 and Figure C-4 for an example of a CREATE command.

RESET

- Removes any (blank) insert lines.
- Deletes any column ruler lines inserted by the COLS line command.
- Cancels any pending line commands.

RIGHT

Scrolls the edit panel to the right-hand side of the displayed data by the amount specified in the **Scroll** field. The RIGHT command is assigned to F11.

SAVE

Saves any changes to the data. If you use the I line command to add lines, any insert lines which are not used, are not saved.

TOP

Scrolls the edit panel to the first page of data. The `***TOP-OF-DATA***` indicator is displayed at the top of the data.

UP

Scrolls towards the top of the data by the amount specified in the **Scroll** field. The UP command is assigned to F7.

Panel Editor Line Commands

Line commands are used to perform the following operations:

- Insert or delete lines
- Repeat lines
- Copy and move lines or overlay lines
- Display a column ruler

Line commands can be entered in either upper or lower case and are translated to upper case before use.

Line commands are entered anywhere within the line command field that prefixes each line. The command can overtype the insert line arrow (->) or in-use indicator (’’’’) that fills the line command area. If a line command is suffixed by a duplication number (for example, **R3**), only blanks can separate the command and its duplication number.

Table C-3. Panel Editor Line Command Summary

Command	Description	Block Command *
A	After	No
B	Before	No
C	Copy	Yes
D	Delete	Yes
I	Insert	No
IB	Insert Before	No
M	Move	Yes
O	Overlay	Yes
Q	Save Block	Yes
R	Repeat	Yes
(	Shift data left	Yes
)	Shift data right	Yes
COLS	Insert column ruler	No

* See *Identifying Blocks of Lines* for a description of block commands.

Identifying Blocks of Lines

If an action such as copy is to be performed on a group of lines, the group is delimited by marking the top and bottom lines of the group with a double command character. Figure C-5 shows an example of a block of lines being copied using CC to designate the top and bottom lines in the block. In this example, lines 2 through 6 are copied after line 7 (before the `***END-OF-DATA***` line).

Figure C-5. Example of a Block of Lines Being Copied

```
LIB: DEVEL   ----------------- MODS : Edit Panel -----------CAPS OFF--NULLS ON-
 COMMAND ===>                                    NAME: PANEL1       SCROLL: HALF
       ***TOP-OF-DATA***                                             L1 C1 74
 ’’’’ This is line 1 of my data
 CC’’ This is line 2 of my data
 ’’’’ This is line 3 of my data
 ’’’’ This is line 4 of my data
 ’’’’ This is line 5 of my data
 CC’’ This is line 6 of my data
 ’’’’ This is line 7 of my data
 B’’’  ***END-OF-DATA***
```

The blocking option can be more convenient than using a duplication number since it is not necessary to know the exact number of lines in the block. The Panel Editor Line Command Summary (Table C-3.) indicates which commands support the blocking option.

A*nn*

Insert copied or moved line(s) after the current line, *nn* times. If *nn* is omitted, the line(s) will be inserted once only.

B*nn*

Insert copied or moved line(s) before the current line, *nn* times. If *nn* is omitted, the line(s) will be inserted once only.

C*nn*

Copy *nn* lines starting from the current line. Enter A or B beside the line of data to which the lines are to be copied after or before, or O*nn* or OO to copy the data over a sequence of lines.

If *nn* is omitted, only the current line is copied. If a block of lines is to be copied, mark the top and bottom lines of the block with CC.

COLS

Insert a column ruler line above the line containing the COLS command. This column ruler can help you align data in specific columns.

Any number of column ruler lines can be displayed. They are ignored when data is saved.

The column ruler line can be deleted by either a RESET command or a D line command.

D*nn*

Delete *nn* lines starting from the current line. If *nn* is omitted, only the current line is deleted. If a block of lines is to be deleted, mark the top and bottom lines of the block with DD.

I*nn*

Insert *nn* blank lines after the current line. If *nn* is omitted, one blank line is inserted.

Lines inserted using the I command are termed *insert lines* and appear with an arrow (->) in the line command area. Once data is added to these lines the arrow is changed to the active line indicator (’’’’). Blank insert lines are ignored when data is saved.

Insert lines can be deleted by either a RESET command or a D line command.

IB*nn*

Insert *nn* blank lines before the current line. If *nn* is omitted, one blank line is inserted.

Lines inserted with the IB command are termed insert lines and appear with an arrow (->) in the line command area. Once data is added to these lines the arrow is changed to the active line indicator (’’’’). Blank insert lines are ignored when data is saved.

O*nn*

Overlay copied or moved line(s) over the next *nn* lines. Blank characters in the target lines(s) are overlaid by the corresponding characters from the source lines. The number of source and target lines need not be the same. If there are more target lines, the source lines are repeated until the target lines are used. If there are more source lines, the extra source lines are ignored.

If *nn* is omitted, it defaults to the number of lines being moved or copied. If a block of lines is to be overlaid, mark the top and bottom lines of the block with OO.

Figure C-6 shows an example of an O line command before execution. Figure C-7 shows the results of the O command.

M*nn*

Move *nn* lines starting from the current line. Enter A or B beside the line of text to which the lines are to be moved after or before, or O*nn* or OO to move the data over a sequence of lines.

If *nn* is omitted, only the current line is moved. If a block of lines is to be moved, mark the top and bottom lines of the block with MM.

Figure C-6. Example of an O Line Command

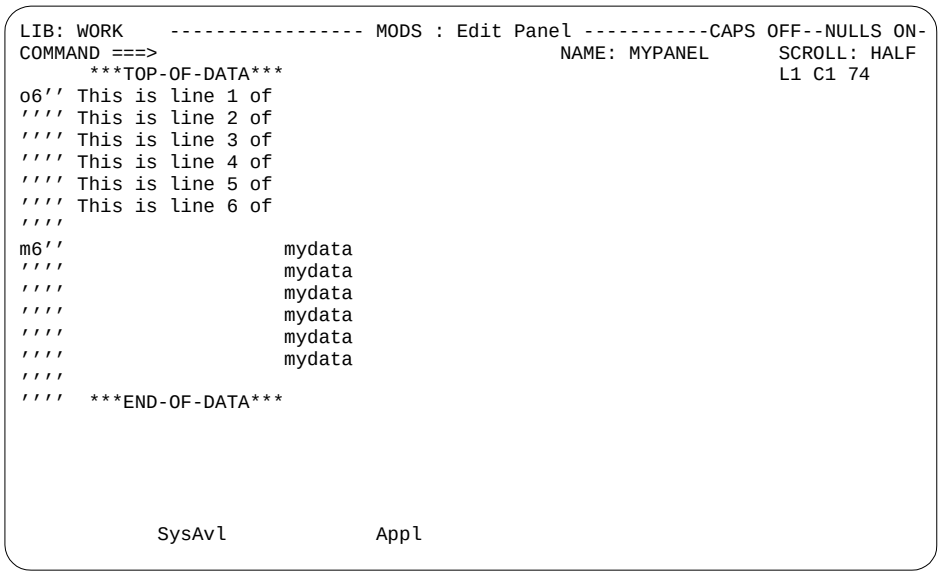
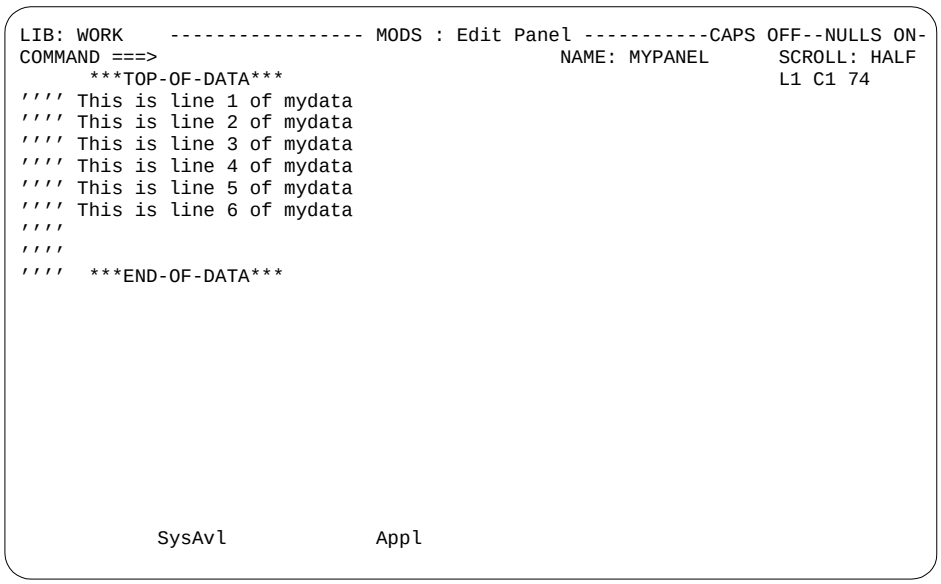


Figure C-7. Example of a Completed O Line Command



Q*nn*

Queue *nn* lines (that is, copy *nn* lines into a temporary storage area for later insertion at different points within the data). If *nn* is omitted, only the current line is queued. If a block of lines is to be queued, mark the top and bottom lines of the block with QQ.

To insert queued data, enter QA or QB to insert the data after or before the current line.

A saved block can be inserted into the data any number of times until the edit session is ended (or until the block is overwritten by another block of data from a subsequent Q command).

R*nn*

Repeat the current line *nn* times. If *nn* is omitted, the current line is repeated once. If a block of lines is to be repeated, mark the top and bottom lines of the block with RR.

)*nn*

Shift the data on the current line *nn* positions to the right. Any data shifted beyond the maximum record length is lost. If *nn* is omitted, the data is shifted one column to the right.

If a block of lines is to be shifted to the right, mark the top and bottom lines of the block with)).

(*nn*

Shift the data on the current line *nn* positions to the left. Any data shifted beyond column one is lost. If *nn* is omitted, the data is shifted one column to the left.

If a block of lines is to be shifted to the left, mark the top and bottom lines of the block with ((.

D

Shorthand Time and Date Formats

CAS data validation supports *shorthand* entry of times and dates.

Time abbreviations and the corresponding values that are returned to the calling procedure by CAS, are summarized in *Table D-1*, *Shorthand Time Formats (HH.MM)* and in Table D-2, *Shorthand Time Formats (HH.MM.SS)*.

Table D-1. Shorthand Time Formats (HH.MM)

Abbreviation	Returns (Edit 4)	Returns (Edit 24)
=	The current time	The current time
+h	Current time + *h* hours	Current time + *h* hours
-h	Current time - *h* hours	Current time - *h* hours
h	0*h*.00	0*h*:00
hh	*hh*.00	*hh*:00
hmm	0*h.mm*	0*h:mm*
hmm	*hh.mm*	*hh:mm*
h.m or *h:m*	0*h.*0*m*	0*h:*0*m*
h.mm or *h:mm*	0*h.mm*	0*h:mm*
hh.m or *hh:m*	*hh*.0*m*	*hh*:0*m*
hh.mm or *hh:mm*	*hh.mm*	*hh:mm*
.m or *:m*	00.0*m*	00:0*m*
.mm or *:mm*	00.*mm*	00:*mm*

Table D-2. Shorthand Time Formats (HH.MM.SS)

Abbreviation	Returns (Edit 20)	Returns (Edit 23)
=	The current time	The current time
h	0*h*.00.00	0*h*:00:00
hh	*hh*.00.00	*hh*:00:00
hmm	*0h.mm.00*	*0h*:*mm:00*
hhmm	*hh*.*mm*.00	*hh*:*mm*:00
hhmms	*hh*.*mm*.0*s*	*hh*:*mm*:0*s*
hhmmss	*hh*.*mm*.*ss*	*hh*:*mm*:*ss*
hh.*mm*.*ss* or *hh*:*mm*:*ss*	If *hh* or *mm* or *ss* are single digits, zero (0) is inserted before the digit. If *hh* or *mm* or *ss* are omitted, they are set to 00—for example, **.2.** becomes **00.02.00**	
+ or - *hh*.*mm*.*ss* or + or - *hh*:*mm*:*ss*	Same rules as above except that the time is added or subtracted from the current time—for example, **+1.5** is 1 hour and 5 minutes from now	

Date abbreviations and the corresponding values that are returned to the calling procedure by CAS are summarized in Table D-3, *Shorthand Date Formats*. All dates are returned in DD-MMM-YYYY format.

Table D-3. Shorthand Date Formats

Abbreviation	Returns
=	Today's date
+d	Today's date + *d* days
-d	Today's date - *d* days
d	The *d* th day of the current month and year
dd	The *dd* th day of the current month and year
ddd	The *ddd* th Julian day of the current year
ddmm or *mmdd*	The *dd*th day of the *mm*th month of the current year (order depends on national language code)
yyddd	The *ddd* th Julian day of the year *yy*
yy.ddd	The *ddd* th Julian day of the year *yy*
ddmmyy or *mmddyy*	The specified date (order depends on national language code)
dd/mm/yy or *mm/dd/yy*	The specified date (order depends on national language code) If *dd* or *mm* or *yy* are omitted, then they default to the current day, month or year (for example, entering //96 on the 3rd January 1992 returns 03-JAN-1996).
day-of-week	The date of *day-of-week* (MON/TUE/WED/THU/FRI/SAT/SUN) in the current week
+day-of-week	The date of *day-of-week* in the following week
-day-of-week	The date of *day-of-week* in the previous week

E

List Panel Attributes

Table E-1 shows a list of panel attributes that can be used to modify the intensity, color and highlighting of data within an entry line.

Table E-1. Panel Attribute Values

Variable	Intensity	Color	Highlight
&$LHATBLBN	Low	Blue	NONE
&$LHATBLGN	Low	Green	NONE
&$LHATBLPN	Low	Pink	NONE
&$LHATBLRN	Low	Red	NONE
&$LHATBLTN	Low	Turquoise	NONE
&$LHATBLWN	Low	White	NONE
&$LHATBLYN	Low	Yellow	NONE
&$LHATBHBN	High	Blue	NONE
&$LHATBHGN	High	Green	NONE
&$LHATBHPN	High	Pink	NONE
&$LHATBHRN	High	Red	NONE
&$LHATBHTN	High	Turquoise	NONE
&$LHATBHWN	High	White	NONE
&$LHATBHYN	High	Yellow	NONE
&$LHATBLBR	Low	Blue	REVERSE
&$LHATBLGR	Low	Green	REVERSE
&$LHATBLPR	Low	Pink	REVERSE

Table E-1. Panel Attribute Values (Continued)

Variable	Intensity	Color	Highlight
&$LHATBLRR	Low	Red	REVERSE
&$LHATBLTR	Low	Turquoise	REVERSE
&$LHATBLWR	Low	White	REVERSE
&$LHATBLYR	Low	Yellow	REVERSE
&$LHATBHRH	High	Blue	REVERSE
&$LHATBHGR	High	Green	REVERSE
&$LHATBHPT	High	Pink	REVERSE
&$LHATBHRR	High	Red	REVERSE
&$LHATBHTR	High	Turquoise	REVERSE
&$LHATBHWR	High	White	REVERSE
&$LHATBHYR	High	Yellow	REVERSE
&$LHATBLBB	Low	Blue	BLINK
&$LHATBLGL	Low	Green	BLINK
&LHATBLPB	Low	Pink	BLINK
&$LHATBLTB	Low	Turquoise	BLINK
&$LHATBLWB	Low	White	BLINK
&$LHATBLYB	Low	Yellow	BLINK
&$LHATBHBB	High	Blue	BLINK
&$LHATBHGB	High	Green	BLINK
&$LHATBHPB	High	Pink	BLINK
&$LHATBHRB	High	Red	BLINK
&$LHATBHTB	High	Turquoise	BLINK
&$LHATBHWB	High	White	BLINK
&$LHATBHYB	High	Yellow	BLINK

F

SOLVE Web File Utilities

The SOLVE Web file system contains all the files used by the SOLVE Web Interface. The Web file system is stored in the MODS file; however, it has no relation to the other MODS components such as the Common Application Services (CAS) functions.

In general, the SOLVE Web file system is accessed only by SOLVE internal servers. For diagnostic purposes, a limited range of SOLVE Web file utilities are available online. This appendix describes the SOLVE Web file utilities.

About SOLVE Web File Utilities

The Web file utilities are accessed through the CAS : Maintenance Menu, by selecting option **W** – Web Files (see Figure F-1). The Web File Utilities : Top Level Directory Summary panel is displayed as shown in Figure F-2.

Figure F-1. CAS : Maintenance Menu

```
SOLVPROD------------------- CAS : Maintenance Menu --------------------$CA010
Select Option ===>

   M   - Menus
   L   - Lists
   P   - Panel Domains
   H   - Help
   MS  - Messages
   T   - Tables
   C   - Criteria
   CM  - Commands
   W   - Web Files
   X   - Exit









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

The Web File Utilities : Top Level Directory Summary panel is a summary of all files present in the SOLVE Web file system.

Figure F-2. Web File Utilities : Top Level Directory Summary

```
SOLVPROD----- Web File Utilities : Top Level Directory Summary ----------------
Command ===>                                                 Scroll ===> PAGE
W3DB9002 7 directories found, containing a total of 396 files.
                                              (S = Display directory contents)
    Directory Name                               # of files
    COMMON                                              54
    INTERNAL                                             6
    JAVA                                                14
    LOGON                                                3
    NETSCAPEUPDA                                         1
    PUBLIC                                              91
    TCPIP                                              227
    **END**









 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap
```

The display fields on the Web File Utilities : Top Level Directory Summary panel are as follows:

Directory Name
Displays the top level or first segment of the full path name of the Web file.

# of files
Displays the total number of Web files in all subdirectories contained in this top level directory.

To display a list of files and subdirectories contained in a top level directory enter **S** beside the directory name. The Web File Utilities : Directory panel for the selected directory is displayed as shown in Figure F-3.

Figure F-3. Web File Utilities : Directory Listing

```
SOLVPROD---------- Web File Utilities : Directory "COMMON" --------------------
Command ===>                                                 Scroll ===> PAGE
W3DB9105 54 files found in  "COMMON" and its subdirectories.
    (S/B=Browse File or Subdirectory, E=Edit File, D=Delete File, I=File Info.)
    File or Subdirectory Name                          Fix Lvl Type   DD
    AlertMonitor                                               Directory
    chgpwd01.esp                                       Install TEXT   MODSTST1
    ContentMenu.esp                                    Install TEXT   MODSTST1
    dataframework                                              Directory
    functions                                                  Directory
    help                                                       Directory
    javachart                                                  Directory
    logoffi.html                                       Install BINARY MODSTST1
    registration                                               Directory
    SelectSolveLinkMultiple.htmlf                      Install TEXT   MODSTST1
    SelectSolveLinkSingle.htmlf                        Install TEXT   MODSTST1
    Solve.esp                                          Install TEXT   MODSTST1
    SolveLinksSingle.htmlf                             Install TEXT   MODSTST1
    SolveMenu.shtml                                    Install TEXT   MODSTST1
    SolveMenuApplet.esp                                Inhouse TEXT   MODSUSR
    SolveMenuApplet.shtml                              Install TEXT   MODSTST1
    SolveMenuLogo.shtml                                Install TEXT   MODSTST1
 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap
```

The display fields on the Web File Utilities : Directory panel are as follows:

File or Subdirectory Name
Displays the file name of an individual Web file, or the name of a subdirectory.

Fix Lvl
Displays the fix level of the Web file.

Install
Indicates that this file is still at the level that was installed from the original product tape.

NZ12345
Indicates that this file was modified by this SOLVE Maintenance Tape fix

Inhouse
Indicates that this file has been modified at your site after installation.

Type (files only)
Indicates whether the data in the Web file is stored in text or binary format.

DD
Displays the DD name of the highest level in the MODS concatenation that this file is present in.

An identically named Web file can exist in more than one level in the MODS concatenation, but is only retrieved from the highest level. In general, Web files installed as directed from SOLVE product or maintenance tapes are installed into the MODSDIS level. Web files modified after installation appear in the MODSUSR level.

The options on the Web File Utilities : Directory panel are described below:

Use Option…	To…
S/B (text files)	Browse the source of the Web file. Only files containing text can be browsed.
S/B (subdirectories)	List the individual files and subdirectories contained in the subdirectory.
E (text files)	Allow editing of the source of the Web file. Only files containing text in lines of less than 256 characters can be edited. An edited Web file, when saved, is placed in the highest level of the MODS concatenation (usually MODSUSR). You should not edit files unless asked to do so by Sterling Software support personnel.
D	Deletes the Web file from the DD level displayed. Only Web files in the highest level (usually MODSUSR) can be deleted.
I (files only)	Display detailed information about the Web file. Information includes the file size and the last updated details. This information is displayed separately for every DD in the MODS concatenation in which this Web file is present.

Glossary

This glossary defines the terms and abbreviations commonly used with SOLVE management services.

It also includes references to terms used in an IBM environment and any equivalent FUJITSU terms.

3270 VDU terminal
: An IBM video display terminal. This is often used to refer to the entire range of 3270 terminals. When followed by a number (for example, 3270-5), a specific model is intended.

370/390
: This is an abbreviation for IBM's System 370 or S/370 architecture. It is often used to indicate any mainframe CPU that implements this architecture.

3705/3725/3745
: An IBM front end communications processor (The Fujitsu equivalent is a CCP or 2806).

9526
: A Fujitsu video display terminal.

ACB (Access method Control Block)
: A control block that links an application program to an access method such as IBM's VTAM or VSAM.

ACB-sharing
: MAI's ability to use a single VTAM ACB for multiple sessions.

Access Security Exit

An installation-provided routine that may be used to replace the SOLVE management services UAMS functions, partially or completely, allowing logon, logoff, and password maintenance requests to be passed to an external security system.

ACF/VTAM (Advanced Communication Facility/VTAM)

IBM's product implementation of SNA's SSCP or CP.

Activity Log

A system-maintained log that records all important activity for use in later problem determination.

Advanced Program to Program Communications (APPC)

An IBM-defined application level protocol which makes use of SNA's LU 6.2.

AIM

Fujitsu's main database/data communications system.

Alternate index

An alternative view of the data contained within a VSAM keyed dataset. The alternate index allows data to be retrieved using an alternate key in addition to the usual access through the primary key.

AOM (Advanced Operation Management)

A facility of SOLVE management services that manages and controls local and remote operating systems.

AOMPROC

The name given to an NCL procedure used to intercept messages from the screening table component of AOM to provide extended message processing.

APF (Authorized Program Facility)

Describes the special authorization level required within the operating system for certain applications.

APPC (Advanced Program to Program Communications)

An IBM-defined application level protocol which makes use of SNA's LU 6.2.

APPL

A VTAM term used to describe the definition that allows an application to use VTAM facilities.

Application Plan

A DB2 term that refers to the control structure used by the database to allocate resources and execute SQL statements.

ASN.1

Abstract Syntax Notation One, defined by ISO 8824, is an abstract syntax used to describe data structures. It is used by Mapping Services to define data structures within SOLVE management services.

Authorized Program Facility (APF)

Describes the special authorization level required within the operating system for certain applications.

Authorization ID

A DB2 term that refers to the ID used by DB2 to control access to a database.

BER (Basic Encoding Rules)

The transfer syntax used by Mapping Services to serialize data for transmission. It is defined by ISO 8825.

BIND

1. A VTAM term describing the action of logically linking one network resource with another network resource.
2. A DB2 term for the process that connects together the application plan and the Database Request Module (DBRM) used by an application.

Broadcast Services

Broadcast Services controls the sending of messages throughout SOLVE systems.

CAF (Call Attach Facility)

A connection technique provided by IBM and used by the EDBS facility of SOLVE management services, to communicate with DB2.

Checkpoint

Refers to a point of synchronization in processing where a unit of work is complete, or partially complete, such as where data is recorded for restart purposes. A point at which information about the status of transmission can be recorded so that it can be restarted later.

CICS (Customer Information Control System)

An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs.

Client

A functional unit that receives shared services from a server.

CNM (Communications Network Management)

IBM term for its SNA management facilities.

CNMPROC

The name given to an NCL procedure used to intercept CNM records received across the VTAM CNM interface by the NEWS component of SOLVE:Netmaster.

Command Partition

A term associated with NPF that describes the group of network resources a user ID is authorized to reference with VTAM commands.

Configuration Management

An ISO/OSI classification of management functions that apply to the ability to set or change operating parameters of the system, to collect and distribute information on their status, to associate names with the entities, and to change the system configuration.

Control Member

A term associated with NPF that describes the list of resource table names applying to a user ID. This control member is referenced in the definition of USERID.

Cross-domain Resource

A VTAM term describing the definition of a Network resource that is owned by a VTAM in another domain.

DBCS (Double Byte Character String)

Refers to a mode of representation of data where each byte of data requires 16 bits rather than 8 bits as required by Single Byte Character String. DBCS is used for the implementation of languages such as Kanji, which is used in Japan.

Deferred Write

A performance option for use with UDBs to minimize I/O by deferring the writing of records.

DEFLOGON

The term used to describe an application entry path to be supported by the EASINET and MAI-FS features. The DEFLOGON command is used to define these paths and their associated text strings.

Dependent Processing Environment

An NCL processing environment which is a child process to another NCL process and hence has its output delivered as input to some other NCL procedure.

DOM (Delete Operator Message)

A non-roll deletable message (NRD) can be deleted from a window only when a DOM is issued.

Domain

1. An SNA term describing a domain that consists of the set of SNA resources controlled by one common control point called an SSCP. In terms of implementation, an SSCP is the host access method (VTAM). An SNA network consists of one or more domains.
2. A VTAM term that describes a logical division of a network. Networks are divided into domains that are associated with the way they are controlled.

Domain ID

Term for a 1-4 character mnemonic used as a unique identifier for a SOLVE system.

Dynamic Allocation

Assignment of datasets to a program at the time the program is executed rather than at the time the job is started.

EASINET

The name of the set of SOLVE management services functions available from SOLVE:Access. EASINET allows idle terminals to be brought under the control of SOLVE management services and to be operated by installation-written NCL procedures that can provide a wide range of *front end* facilities to end-users of the network.

ER (Explicit Route)

The physical path between two network nodes. (SNA)

ESDS (Entry Sequenced Data Set)

A non-keyed VSAM dataset whose records are stored and retrieved in sequential order, and new records added to the end of the data set.

ESF (Expert Systems Foundation)

A rule-based facility that automates the handling of messages captured by AOM.

Exception

The result of a service request that did not complete successfully. See *Reply* and *Response*.

Exit

An installation-written routine that can be driven from a point within a program to provide data to the program, or perform additional processing relevant to that installation's specific requirements.

Extended Datastream

A 3270 datastream containing fields that utilize color and extended highlighting capabilities of the terminal.

External Database Connection

A term used to refer to connections from SOLVE management services to the database which use the Call Attach Facility (CAF) to connect to DB2. NCL connections are connections to DB2 that utilize the management services connection. See also *Management Services Connection* and *NCL Connection*.

External Database ID

The ID of the external database as it is known to the operating system.

Fault Management

An ISO/OSI classification of management functions relating to the detection, collection, diagnosis, testing, correction, and prevention of network and system faults and error conditions.

FID

A File ID under DOS.

FSP

A Fujitsu operating system.

Function Key

A key on a terminal's keyboard which causes a panel to be completed. In the case of 3270, this term also applies to the ENTER key (the same behavior can be achieved on a 6530).
Previously called Program Function (or PF) keys.

IMS (Information Management System)

IBM's database/data communication (DB/DC) system that can manage complex databases and networks.

Initiator

The component of FTS which schedules transmission of a dataset.

INMC (Inter-Management Services Connection)

This facility allows SOLVE systems running in a network containing multiple CPUs to communicate with each other, providing general-purpose data transfer between CPUs within the network.

INMC/EF (INMC/Extended Function)

INMC/EF provides the capability for up to sixteen sessions between any pair of SOLVE systems. In appropriate systems, these sessions can traverse different physical network paths, thus increasing throughput. This component also provides additional link security and management facilities.
In releases prior to Version 3.0, INMC/EF was a separate component. At release 3.0 it became a part of INMC.

Inter-Management Services Connection (INMC)

See *INMC (Inter-Management Services Connection).*

I/O

Input/Output

ISPF (Interactive System Productivity Facility)

An IBM product that is the functional equivalent of Fujitsu's PFD.

KSDS (Key Sequenced Data Set)

A VSAM dataset whose records are directly accessed by a user-supplied key.

Link

A term used to describe a logical connection between two or more SOLVE systems. See also *INMC (Inter-Management Services Connection)*.

Logical Unit (LU)

The point of access for any user to an SNA network. See also *LU*.

LOGMODE

A VTAM term used to describe a table entry that defines the characteristics and protocols of a terminal.

Logoff

A request by an LU that it be disconnected from a VTAM application program.

Logon

A request by an LU that it be connected to a VTAM application program.

Logon-path

A path through which users of EASINET and MAI-FS gain access to other VTAM application programs. Paths are defined and controlled by the DEFLOGON command.

LOGPROC

The name given to an NCL procedure used to intercept messages destined for the SOLVE management services activity log.

LSR (Local Shared Resources)

A technique for buffering I/O to VSAM files called LSR pools. NCL supports this type of processing for User Databases (UDBs).

LU (Logical Unit)

SNA introduced the concept of the logical unit (LU). The LU is a type of SNA network-addressable unit (NAU) that provides protocols for end users to gain access to the network and to the functional components of the LUs. See also *Logical Unit*.

LU0

An unconstrained SNA protocol that allows implementers to select any set of available protocol rules, as long as the two LUs are able to communicate with each other successfully according to the rules chosen. Therefore, all LU types are an implementation of LU Type 0.

LU1

A line-by-line or typewriter type terminal (for example 3767, 3770), using SNA protocols.

LU2

A 3270 type terminal using SNA protocols.

LU3

LU Type 3 was implemented to support printers with a different data stream format. LU Type 3 is used by printers attached to an IBM display cluster controller.

LU4

LU Type 4 was implemented so that office system products could transfer documents. LU Type 4 is used by banking devices.

LU6.2

A protocol that serves as a port into an SNA network. LU6.2 defines a specific set of services, protocols, and formats for communication between logical processors. LU6.2 provides presentation services for presentation of data to the end user, transaction services for performing transaction processing on behalf of the end user and LU services for managing the resources of the LU.

LU7

An SNA protocol that is used by word-processing devices.

Macro

A single source language statement that expands into a series of statements when the macro is assembled or compiled.

MAI (Multiple Application Interface)

A facility of SOLVE:Access which is used to provide sessions with any number of other VTAM application programs from one SOLVE terminal.

MAI/EF (MAI/Extended Function)

A SOLVE:Access facility which provides an extension to NCL to support session scripts which allow an NCL procedure to control the application and terminal session flow. MAI/EF also includes the Session Replay Facility (SRF) which allows recording and replay of session scenarios.

MAI-FS (MAI/Full Screen)

A SOLVE:Access facility that allows a single SOLVE terminal to be used to provide full screen access to any number of other applications. The user may 'jump' from one application to another using designated 'jump' keys on the keyboard, or special command strings.

MAI-OC (MAI/Operator Control)

A SOLVE management services facility that allows an operator in OCS to have LU Type-1 sessions with many other applications from an OCS window. When used in conjunction with MSGPROC NCL procedures this can provide automated central monitoring and operation of multiple applications from the one operator console.

Management Services

This the central core of functions and service routines within SOLVE. It supports all of the SOLVE products.

Management Services Connection

A SOLVE management services connection is a communication link from SOLVE management services to the external database. It is started, stopped, defined, and deleted using the EDB command. A management services connection must be defined and started before opening an NCL connection. See also *NCL Connection*.

MDO (Mapped Data Object)

Any data item that can be represented as a continuous string a bytes in storage.

Message partition

A term associated with NPF that describes the group of network resources for which a user ID will receive unsolicited (PPO) VTAM messages.

Modify Interface

A means of communicating with an application program from the system console in OS/VS systems. The MODIFY command, abbreviated to F, avoids having an outstanding REPLY at the system console. SOLVE management services supports the use of the MODIFY command.

MSGPROC

An NCL procedure used to intercept and process messages destined for a user's Operator Console Services (OCS) window.

MSP

An operating system for large scale Fujitsu systems.

MVS (Multiple Virtual Storage)

IBM's major operating system. Fujitsu's functional equivalent is MSP.

NAU

See *Network Addressable Unit (NAU).*

NCL (Network Control Language)

The interpretive language that allows logical procedures (programs) to be developed externally to SOLVE management services and then executed by SOLVE management services on command. NCL contains a wide range of logic, built-in functions and arithmetic facilities which can be used to provide powerful monitoring and automatic control functions.

NCL Connection

An NCL connection is a communication link from an NCL procedure to the external database. An NCL connection is opened and closed with the &EDB verb. A SOLVE management services connection must be defined and started before opening an NCL connection. See also *Management Services Connection*.

NCL Procedure

A member of the procedures dataset comprising NCL statements and SOLVE or VTAM commands. The NCL statements and other commands are executed from an EXEC or START command specifying the name of the procedure.

NCL Process

The NCL task that is invoked, usually by a START command to execute one or more associated procedures. Each NCL process has a unique NCL process identifier.

NCL Processing Environment

Provides the internal services and facilities required to execute NCL processes for the user, from its associated SOLVE window.

NCL Processing Region

All users (real or virtual) have an NCL Processing Region associated with their user ID while logged on. This region provides all of the internal services needed to allow the user to have processes executed on their behalf. There may be a maximum of two active NCL environments in a user's NCL region.

NCLID

A 6 digit NCL process identifier which is unique within the system. It is used to identify a process for the purpose of communicating with that process.

NCL/EF (NCL/Extended Function)

An extension to NCL which provides a relational database facility that can be used as a repository for applications running within a SOLVE system. Full update capabilities, including scans with extensive boolean logic, are provided.

NCP (Network Control Program)

This resides within and controls the operation of a communications controller. The NCP communicates with VTAM.

NCS (Network Control Services)

A facility of SOLVE:Netmaster that provides full screen displays and navigation of the network in SOLVE management services.

NET/MAIL

A SOLVE product that provides a personal electronic mail facility for network users.

Network Addressable Unit (NAU)

In SNA, a logical unit, a physical unit, or a system services control point. The NAU is the origin or destination of information transmitted by the path control network. Synonymous with network accessible unit.

Network Control Language

See *NCL*.

NEWS (Network Error Warning System)

A facility of SOLVE:Netmaster which is used to provide network error and traffic statistics and error alert messages.

NMINIT

The NCL procedure automatically executed after system initialization has completed. It cannot contain commands that require VTAM facilities as it is executed before the primary ACB is opened. The procedure name can be changed by the installation.

NMREADY

The NCL procedure automatically executed once system initialization has completed. It can contain commands that require VTAM facilities as it is executed after the primary ACB is opened. Procedure name can be changed by the installation.

Node

A connection point in a communications network.

Non-Roll Delete Message

See *NRD (Non-Roll Delete) Message*.

NPF (Network Partitioning Facility)

A facility of SOLVE management services that allows the range of resources which an operator can influence to be denied.

NPF Control Member

A member of the NPF dataset member which defines a list of member names that are to be the resource tables for the associated user ID.

NPF Resource Table

A member of the NPF dataset that defines a group of network resource names. The resource names can be defined specifically or generically using wildcard characters. A resource table is pointed to by a control member.

NRD (Non-Roll Delete) Message

A message that will not roll off an OCS window display until explicitly deleted. See *DOM*.

NT 2.1

Node Type 2.1. A node in an SNA network. It implements a peer-to-peer protocol and allows greater dynamics in network configuration, greater independence in session set up between partner LUs and reduced definitions.

NTS (Network Tracking System)

A facility of SOLVE:Netmaster used to provide SNA session monitoring, dynamic online network tracing, accounting, and response time information in conjunction with diagrammatic representations of session partners.

OCS (Operator Console Services)

A facility of SOLVE management services that provides general operational control and an advanced operator interface to VTAM for network management.

OS/MVS (Operating System/Multiple Virtual Storage)

see MVS.

OSI (Open Systems Interconnection)

A set of ISO standards for communication between computer systems.

Packet

The logical unit of transmission in a network.

Panel Maintenance

A facility of SOLVE management services which allows users to generate and modify panel definitions used for presentation purposes by NCL procedures. In releases prior to Version 3.0, this function was known as Edit Services.

Panel Skip

The ability to chain menu selection requests together without having to display intermediate selection panels.

Password

A 1 to 8 character string chosen by a user and linked to their user ID for security purposes. To gain access to the system a user must enter both their defined user ID and its associated password.

PDS (Partitioned DataSet)

A type of dataset format that supports multiple individual members in the one physical dataset. Equivalent to the Source Statement Library in VSE/SP systems.

Peer-to-Peer Management
A non-hierarchical heterogeneous network management system.

PFK (Program Function Key)
See *Function Key.*

Physical Unit (PU)
The control unit or cluster controller of an SNA terminal. The part of a control unit or cluster controller which fulfils the role of an SNA-defined physical unit.

PIU (Path Information Unit)
An SNA packet.

PLU (Primary Logical Unit)
Relates to SNA. A type of LU that is usually used by the application programs in a host. It refers to the BIND sender for a session.

PPI (Program to Program Interface)
PPI is a general-purpose facility which allows programs, written in any language, to exchange data.

PPO (Primary Program Operator)
A VTAM term that describes a facility of VTAM that allows unsolicited network messages to be delivered to an application program, such as SOLVE management services, for processing.

PPOPROC
The name given to the NCL procedure used to intercept unsolicited VTAM (PPO) messages.

Preload
A term applied to NCL procedures which are loaded into the system before being required, to improve system performance.

Primary and Secondary
(SNA) Primary and secondary are SNA terms for describing the LU's role when the session is established. The primary LU sends the BIND request that causes the session to be established, and the secondary LU receives the BIND request. Rules defined in the BIND request determine which of these is the first speaker in the exchange of information.

Print Services Manager (PSM)
PSM is a facility of SOLVE management services which simplifies the control of the physical printing of reports on JES or network printers.

Program to Program Interface (PPI)
PPI is a general-purpose facility which allows programs, written in any language, to exchange data.

PSM

See *Print Services Manager (PSM).*

PU

(SNA) Physical Unit. Each node (a logical grouping of hardware) in an SNA network is addressed by its PU. There are 4 types of nodes or PU in an SNA network: PU-T5, PU-T4, PU-T2, PU-T1. See *PU Type* x. A PU is a type of NAU. (See *Network Addressable Unit (NAU)*.

PU Type 1

(SNA) A type of Physical Unit or Node in an SNA network. Consists of a terminal (such as an IBM 3278).

PU Type 2

(SNA) A type of Physical Unit or Node in an SNA network. Consists of a cluster controller (such as an IBM3274, 3276, 3770 or 3790).

PU Type 4

(SNA) A type of Physical Unit or Node in an SNA network. Consists of a communications controller (such as an IBM 3704, 3705, 3725 or 3745).

PU Type 5

(SNA) A type of Physical Unit or Node in an SNA network. Consists of a host computer system (such as an IBM30*xx*, S/370 or 43*xx*, running VTAM or sometimes VCAM).

Remote Operator Facility (ROF)

A facility of SOLVE management services that allows an operator to sign on to a remote location, execute commands and have the results returned.

Remote Terminal Access Method (RTAM)

A facility that controls operations between remote terminals and job entry subsystems.

Reply

The information returned to a directive as a result of a request. This information may be either a response or an exception, together with appropriate arguments. See *Exception* and *Response*.

Request

The invocation of a directive, together with appropriate arguments. See *Exception* and *Response*.

Request Unit

(SNA) A message unit that contains control information such as a request code, or function management headers, end-user data, or both.

Reserved word

The term given to a token that will terminate an input expression if found unquoted outside the topmost parenthesis level. In the context of a particular verb statement the verb keywords are reserved words. A reserved word has special meaning for the current statement only, and different statements have different reserved words. Note that there are no words that are reserved throughout NCL.

Resource Table

A term associated with NPF that describes a list of resource names or generic resource names that define a command or message partition.

Response

The success result of a service request. See *Exception* and *Reply.*

Response Unit

(SNA) A message unit that acknowledges a request unit.

Return Code

A code returned from the system that indicates the success or failure of the task performed.

ROF

See *Remote Operator Facility (ROF).*

RTAM

See *Remote Terminal Access Method (RTAM).*

RTM (Response Time Monitor)

A facility provided by IBM's 3*x*74 control units to monitor end-user response times. NEWS can interpret this data.

RTM (Response Time Measurement)

Measurement of the time which passes between the user starting an action (by pressing a key) and the response appearing on the screen.

RU (Request/Response Unit)

(SNA) A generic term for a request or response unit.

SAW (Session Awareness Data)

A type of network management data supplied by VTAM and processed by NTS.

Security Initialization Unit

A hardware device that creates and loads encrypting codes, also known as *keys*, for your computer system.

Sequence Number

A number assigned to each message exchanged between a VTAM application program and a logical unit. Values increase by one throughout the session, unless reset by the application program using an STSN or CLEAR command.

Server

A process designed to serve the data to a client, or request process, for one or more users.

Session Name

A name assigned to a workstation or session to permit it to receive messages or share resources.

SIS (Screen Image Services)

A part of the MAI/EF facility of SOLVE:Access which provides the ability to record screen images for later retrieval or to send to another user.

SLU (Secondary Logical Unit)

(SNA) A type of LU that is usually used by the end-users at the terminals or by programs which reside in the peripheral node.

SMF (System Management Facility)

An optional control feature of OS/VS that provides the means for gathering and recording information that can be used to evaluate system usage.

SNA (Systems Network Architecture)

This term describes the logical structure, formats, protocols, and operational sequences for transmitting communication data through the communication system (Fujitsu equivalent is FNA). A set of standards that allows the integration of all the different IBM hardware/software products into a universal network. Introduced in 1974.

SOLVE

This is the name of a suite of systems management products which assist in the management and operation of an organization's information processing environment.

SOLVE:Netmaster

A SOLVE product which addresses the problems of Network Management. It comprises NCS, NTS and NEWS.

SRF (Session Replay Facility)

A part of the MAI/EF facility of SOLVE:Access which provides the ability to record and playback terminal session scenarios.

Structured field

Representation of user ID attribute information exchanged between SOLVE management services and its security exit.

Subtask

A unit of work whose environment is established by a main task, but has its own TCB, and is displaceable by the operating system.

SYSPARMS

System parameters—values that affect some SOLVE system capabilities. Some SYSPARMS can be modified dynamically.

Thread

A unit of work running under the control of an application program.

Timestamp

The instant of time at which the information described by a data item was valid.

TSO (Time Sharing Option)

Allows terminal operators to interact directly with computer resources and facilities. Used mainly by application and system programmers. (Fujitsu equivalent is TSS).

UAMS (Userid Access Maintenance Sub-system)

The security component of SOLVE management services that supports the definition of authorized users and their associated function and privilege levels.

UCS (Universal Character Set)

A printer feature that permits the use of a variety of character sets.

UDB (User Data Base)

1. UDB file access method layer allowing file access from NCL.
2. A term used to identify VSAM datasets to which NCL procedures may have access using the &FILE verb (GET, PUT, ADD, and DEL options).

UDM (UnDeliverable Message)

A term that applies to the Network Partitioning Facility (NPF) of SOLVE management services. It describes a message that cannot be directed to a terminal operator partitioned for the resource to which the message refers, or a message that does not apply to a specific resource.

User ID

Defines the function and privilege level to which a specific user is entitled when they sign on to the system. It is associated with a secret password to prevent use by unauthorized personnel. This definition is stored in the UAMS dataset or on an external security system.

USS (Unformatted System Services)

A VTAM term that describes a facility that translates an unformatted command such as LOGON or LOGOFF, into a field formatted command for processing by formatted system services. Applies to terminals before connection to an application.

Verb

The term given to a stand-alone statement in an NCL program. NCL *verbs* cause actions to occur. There are different types of verbs, some that dictate the flow of processing and logic, others that fetch information for the procedure to process and others that cause data to flow to external targets.

VFS (Virtual File Services)

The VSAM dataset, used by many facilities as a database.

VM (Virtual Machine)

A superset operating system that allows other operating systems to run as if they each had their own machine.

VM/ESA

An enhanced version of VM that supports 31-bit addressing.

VM/XA (Virtual Machine/Extended Architecture)

An extended version of VM.

VOS3

A Hitachi operating system comparable to IBM's MVS.

VR

(Session) Virtual Route.

VSAM (Virtual Storage Access Method)

A method for processing data files that utilizes relative, sequential, and addressed access techniques. Conceptually identical to ENSCRIBE.

VSE (Virtual Storage Extended)

An operating system for small IBM systems.

VTAM (Virtual Telecommunications Access Method)

A suite of programs that control communication between terminals and application programs.

Wildcard

The term used to describe the character used (usually an asterisk) when defining resources generically — no specific matching character is required in the wildcard character position.

XSP

A Fujitsu operating system. Successor to FSP.

XNF

A Hitachi network access method for OSI networks.

Index

Symbols

#ALIAS
    panel control statement 6-22
#ERR panel control statement 6-29
#FLD
    panel control statement 6-19
$CACALL 2-5
    feedback codes 18-4
    invoking 18-1, 18-3
        BUILD CRITERIA 18-5, 18-7
        BUILD FKA 18-10
        BUILD MESSAGE 18-13, 18-15
        DISPLAY DATA 18-17
        DISPLAY HELP 18-21
        DISPLAY LIST 18-23
        DISPLAY MENU 18-26
        DISPLAY MESSAGE 18-28
        EDIT DATA 18-29
        EXECUTE COMMAND 18-33
        LOAD COMMAND 18-36, 18-39, 18-41
        NAVIGATE PDOMAIN 18-43, 18-47
    return codes 18-4
$EDMENU procedure 6-3
% field character
    (high intensity, protected) 6-20
&$CMPARMS
    command definition 14-4
(command
    panel editor C-16
    text editor B-5
) command
    panel editor C-16
    text editor B-5
+ field character
    (low intensity, protected) 6-20
.AT macro 10-30
.BX macro 10-32
.CE macro 10-34
.CH macro 10-35
.CM macro 10-36
.CP macro 10-37
.CT macro 10-39
.DE macro 10-41
.LI macro 10-43
.LN macro 10-45
.MU macro 10-46
.NP macro 10-49
.OP macro 10-50
.PH macro 10-52
.RA macro 10-53
.SH macro 10-54
.SP macro 10-55
.TI macro 10-56
= command 14-5
? prompted fields 2-11

_ field character
(high intensity, unprotected) 6-21

A

A command
panel editor C-13
text editor B-1
Abbreviated Value
validation table entry 12-18
about maps 25-1
access to MODS 3-5
Action
command definition 14-4
menu definition 7-6
valid commands for defining 14-4
action
$CACALL 18-1
action list 2-7
Active
validation table entry 12-18
Add Allowed?
list definition 8-8
Add as an Alias?
help file 10-4
adding a map definition 25-5
Administration
accessing 3-4
alias help file
defining 10-25
ALL command
text editor B-6
APAR tapes
applying 17-6
API 2-5
application
defining 2-3
Application Definition 2-3
application definition 4-1
adding 4-3
browsing 4-4
copying 4-5
defining 4-2
deleting 4-4
listing 4-5
updating 4-4
Application ID 4-2
criteria definition 13-4, 13-5
validation table definition 12-4, 12-5
Application Level Help File
adding 10-5
browsing 10-6
copying 10-7
deleting 10-7
listing 10-9
printing 10-11, 10-15, 10-20, 10-24
updating 10-7
viewing 10-11
Application Register 2-3, 4-1
accessing 3-3
APPLID 4-2
attribute byte
panel definition 6-20

B

B command
panel editor C-13
text editor B-2
blink
help text 10-30
BOTTOM command
panel editor C-5
brightness
help text 10-30
browse facility
CAS 2-13
browsing a map definition 25-10
browsing ASN.1 source code for a map
25-10

C

C command
panel editor C-13
text editor B-2
cache
reset for list 8-20
CANCEL command
panel editor C-5
CAPS command
panel editor C-5
CAS
Application Definition List (page 1)
4-5
Application Definition List (page 2)
4-6
Application Definition Panel 25-5

CAS (*continued*)
Application Level Help Definition Menu 10-4
Application Level Help Definition Panel 10-5, 10-6
Command Definition Menu 14-2
commands 2-12
control file 2-17, 17-2
criteria 2-12
Criteria Definition List (page 1) 13-9
Criteria Definition List (page 2) 13-9
Criteria Definition Menu 13-3
Criteria Definition Panel 13-5
Criteria Text Panel 13-7
help 2-8
List Definition List (page 1) 8-16
List Definition List (page 2) 8-16
List Definition List (page 4) 8-18
List Definition Menu 8-5
List Description Panel 8-6
List Format Panel 8-13
lists 2-6
log 17-16
maintenance functions 2-4
Maintenance Menu 3-4
Menu Definition List Panel 7-9
Menu Definition Menu 7-2
Menu Description Panel 7-3
menus 2-6
Message Definition List 11-8
Message Text/Explanation panel 11-3
messages 2-10
Panel Domain Definition List (page 1) 9-6
Panel Domain Definition panel 9-4
Panel Domain Element Definition List (page 1) 9-10
Panel Domain Element Definition List (page 3) 9-11
Panel Domain Element Definition panel 9-13
panel domains 2-8
programming interface 2-5, 18-1
Select Alias Help List 10-26
Select Alias Help Panel 10-25
Table Definition Menu 12-5
Table Entry Definition List (page 1) 12-20
Table Entry Definition List (page 2) 12-21
Table Entry Definition Menu 12-14, 12-15, 12-16
CAS (*continued*)
Table Entry Definition Panel 12-17
text editor 2-13
Window Level Help Definition Menu 10-17
CHANGE command
panel editor C-6
text editor B-7
class
$CACALL 18-1
CMD command 14-4
color
help text 10-30
terminals
panel error displays 6-29
COLS command
panel editor C-13
text editor B-2
command definition
adding 14-3
browsing 14-8
copying 14-9
deleting 14-8
listing 14-5
reloading 14-9
updating 14-8
valid Actions
CMD 14-4
DISCONN 14-4
EXEC 14-4
HELP 14-4
KEYS 14-4
LOCK 14-4
NOTEPAD 14-4
PASSWORD 14-4
PQUEUE 14-4
PSKIP 14-5
RETRIEVE 14-5
SPLIT 14-5
START 14-5
SWAP 14-5
WHERE 14-5
Command ID 14-2, 14-3, 14-7
commands 2-12
formatting help files 10-27
panel editor C-1
text editor B-1
Comment Line
list definition 8-13

comments
command definition 14-3
criteria definition 13-6
in help files 10-36
Common Application Services Maintenance
accessing 3-3
compiling a map 25-2
compiling a map's ASN.1sourcecode 25-11
components
help 2-8
validation tables 2-11
control file 2-17, 17-2
accessing 17-4
apply-the-difference 17-32
browsing 17-36
comparing 17-26
concatenation 2-18
copying components between files 17-8
deleting components from 17-23
listing 17-40
log 17-2
moving components between files 17-18
searching 17-38
searching, considerations 17-39
control statements, for panels 6-17
Copies, printing the log 17-18
Copy All Matching Panels option 6-15, 16-4
COPY command
panel editor C-7
copying a map definition 25-11
CREATE command
panel editor C-7
creating and maintaining maps 25-2
criteria 2-12
definition 13-3
adding 13-4
browsing 13-11
copying 13-12
deleting 13-12
listing 13-8
menu 13-3
updating 13-11
exit 13-6
maintaining 13-1
Criteria Appl
element definition 9-14
list definition 8-10
Criteria Name 13-4, 13-5
element definition 9-14
list definition 8-11
Criteria Type 13-4, 13-5
element definition 9-14
list definition 8-10
Criteria Userid
element definition 9-14
list definition 8-11

D

D command
panel editor C-14
text editor B-2
Data Source
list definition 8-8, 13-6
database 2-17
Dataset Name
control file 17-8
panel library definition 16-7
date, shorthand entry D-1
DD Name
panel library definition 16-7
Default Mnemonic
list definition 8-8
defining a map 25-4
deleting a map definition 25-11
Description
command definition 14-3
criteria definition 13-5
DISCONN command 14-4
disp control file 17-9
display attributes
help files 10-30
display fields
defining 6-20
Display User Info Box?
menu definition 7-4
domain 2-8, 9-1
Domain Name 9-4, 9-5
Domain Type 9-3, 9-5
DOWN command
panel editor C-9

E

edit numbers
    validation of input fields 18-48
edit permission
    panel library 6-5, 16-7
Edit Services 6-1, C-1
Edit type
    validation table 12-7
editor 2-13
    commands B-1, C-1
element 2-8
element definition
    adding 9-12
    browsing 9-15
    copying 9-15
    deleting 9-15
    editing 9-16
    listing 9-9
    updating 9-15
    viewing 9-15
Element Name 9-13
Eligibility Test
    element definition 9-14
End Column
    help file 10-18
End Row
    help file 10-17
Entry Line
    list definition 8-14
Entry Message Length 8-9
Entry Message Position
    list definition 8-9
error display (#ERR statement) 6-29
Exclusive
    element definition 9-14
EXEC command 14-4
exit
    criteria 13-6
    list 8-8
    validation 12-7
Exit Name
    criteria definition 13-6
exit procedure
    criteria 22-1
    list 21-1
    table entry validation 23-1

F

feedback codes, $CACALL 18-4
field character panel definition
    defining (#FLD) 6-19
field characters
    defaults
        % (high intensity, protected) 6-20
        + (low intensity, protected) 6-20
        _ (high intensity, unprotected) 6-21
    panel definition 6-19
        character mode 6-19
        hexadecimal mode 6-19
Field description
    validation table 12-7
Field help file 10-8
Field Level Help File
    adding 10-22
    browsing 10-22
    copying 10-23
    deleting 10-23
    listing 10-24
    updating 10-23
    viewing 10-24
Field Name
    help file 10-21
    validation table 12-5
    validation table entry 12-16
field types, panel definition 6-20
    INPUT 6-20
    NULL 6-20
    OUTPUT 6-20
    SPD 6-20
File ID
    control file 17-8
    panel library definition 16-6
FIND command
    panel editor C-9
    text editor B-7
FIRST command
    text editor B-7
FLOW command
    text editor B-7
formatting help files 10-27
From Element
    path definition 9-19
FSM command
    panel editor C-9

Full Value
validation table entry 12-17, 12-18
function key area
defining for panels 6-26
Function Level Help File 10-8
adding 10-14
browsing 10-14
copying 10-15
deleting 10-15
listing 10-16
updating 10-15
viewing 10-16
Function Name
help file 10-13, 10-17, 10-21

G

Get All Entries? list definition 8-8
group list definition 8-5, 8-7

H

HALF command
panel editor C-9
heading list definition 8-14
Help
display attributes 10-28
element definition 9-14
help 2-8
on messages 2-10, 11-4
Help Alias
defining 10-25
HELP command 14-4
Help Description 10-6, 10-26
help files
editing and formatting 10-27
help hierarchy 2-9
help index
defining 10-12
help macros 10-27
.AT (define a character's attributes) 10-30
.BX (draw a box around text) 10-32
.CE (draw a box around text) 10-34
.CH (center a heading) 10-35
.CM (add a comment) 10-36
.CP (copy a help file) 10-37
.CT (copy a help file) 10-39
help macros (*continued*)
.DE (display text for terminal type) 10-41
.LI (display text for licensed feature) 10-43
.LN (draw a line) 10-45
.MU (define a menu) 10-46
.NP (begin a new page) 10-49
.OP (display text for operating system) 10-50
.PH (define a primary heading) 10-52
.RA (remove a character's attributes) 10-53
.SH (define a sub-heading) 10-54
.SP (skip lines) 10-55
.TI (define a title line) 10-56
Help Name
criteria definition 13-6
list definition 8-8
help text
editor 10-27
variables 10-30
help tutorial, defining 10-12
highlighting support
Panel Services 6-31, 6-39
Hold?
printing the log 17-18

I

I command
panel editor C-14
text editor B-2
IB command, panel editor C-14
IMFLD
edit type for validation tables 12-7
IMREC
edit type for validation tables 12-7
INFO/MASTER validation tables
category 12-8
description field 12-8
field 12-8
input fields
defining
menus 7-8
panels 6-20
internal validation 6-20
insert lines, panel editor C-14
intensity, help text 10-30
internal validation (error display) 6-29

Internet site, NMD A-2

K

Keep?
    printing the log 17-18
KEYS command 14-4

L

LAST command, text editor B-7
LC command, text editor B-2
LEFT command, panel editor C-10
Length
    validation table definition 12-9
library
    member 6-2
    panels 6-2
    selection list 6-5
list 2-6
    defining 8-4
    definition 8-1
        adding 8-6
        browsing 8-19
        deleting 8-20
        listing 8-15
        updating 8-20
    efinition
        copying 8-20
    entry line 8-4
    exit 8-8
    reset cache 8-20
    service procedure 8-7
    types
        action 2-7
        multiple select 2-7
        numbered (”pick”) 2-7
        single select 2-7
List Format 8-13
List Name 8-5, 8-7, 8-10
List Type 8-5, 8-7, 8-10
listing map definitions 25-6
Load table?
    validation table definition 12-8
loading a map 25-3
LOCK command 14-4
Log
    accessing 17-16
    browsing 17-16
    clearing 17-18
    control file 17-10
    printing 17-17

M

M command
    panel editor C-14
    text editor B-2
maintaining criteria
    about criteria 13-1
    criteria exit 13-2
    data source 13-2
    exit parameters 13-2
    run time panel 13-2
    substitution of variable data 13-3
maintaining map definitions 25-6
maintaining maps
    SOLVE map library structure 25-1
Mapping Services
    Abstract Syntax Notation One 24-1
    accessing 3-4
    ASN.1 24-1
        Compiler’s Interpretation 24-9
        Type assignments 24-2
    available types
        ADB 24-40
        ANY 24-40
        BIT STRING 24-27
        BOOLEAN 24-25
        CHOICE 24-24
        ENUMERATED 24-34
        EXTERNAL 24-32
        GeneralizedTime 24-38
        GeneralString 24-40
        GraphicString 24-38
        HEX STRING 24-30
        IA5String 24-37
        INTEGER 24-25
        NULL 24-31
        NumericString 24-35
        OBJECT IDENTIFIER 24-31
        ObjectDescriptor 24-32
        OCTET STRING 24-29
        PrintableString 24-35
        REAL 24-32
        SEQUENCE 24-22
        SEQUENCE OF 24-23
        SET 24-20

Mapping Services (*continued*)
available types (*continued*)
SET OF 24-21
TeletexString 24-36
UTCTime 24-37
VideotexString 24-36
VisibleString 24-39
Data Behavior 24-19
Data Interchange Between Open Systems 24-5
Data Tagging 24-12
Defining the logical structure of data 24-2
Defining the Physical Structure of Data 24-4
Map Components 24-10
Map Definitions List (page 1) 25-7
Map Definitions List (page 2) 25-7
Map Definitions List (page 3) 25-8
Map Definitions List (page 4) 25-8, 25-10, 25-12
Map Definitions List (page 5) 25-12
Map Source Definitions 24-6
Mapping Directives 24-14
NCL Reference 24-19
Primary Menu 25-4
Referencing Logical Data Structures from NCL 24-4
Type Checking 24-19
Type Description and Formats 24-18
mapping services primary menu 25-4
maps
about maps 25-1
adding a map definition 25-5
browsing ASN.1 source code for a map 25-10
browsing map definitions 25-10
compiling a map 25-2
compiling a mapsASN.1sourcecode' 25-11
copying a map definition 25-11
creating and maintaining maps 25-2
defining a map 25-4
deleting a map definition 25-11
listing map definitions 25-6
loading a map 25-3
maintaining map definitions 25-6
maintaining maps 25-1
mapping services primary menu 25-4
printing a map definition 25-11
SOLVE map library structure 25-1
updating a map definition 25-10
viewing a map structure 25-12
MARGINS command, text editor B-8
Max abbreviation length
validation table 12-8
MAX command
panel editor C-10
Max description length
validation table 12-8
Max full value length
validation table 12-8
MDOS
Application ID List 17-11
member
of panel library 6-2
member of panel library 6-2
menu 2-6
input fields, maintaining 7-8
maintaining definitions 7-1
maintenance menu 7-2
options, defining 7-6
primary 7-4
service procedure 7-4
menu definition
adding 7-3
browsing 7-10
copying 7-11
deleting 7-11
in a help file 10-46
listing 7-8
updating 7-10
viewing 7-11
Menu input field 7-7
Menu Input Field Attributes 7-4
Menu Number 7-2
menus
CAS
Application Level Help Definition Menu 10-4
Command Definition Menu 14-2
Criteria Definition Menu 13-3
List Definition Menu 8-5
Maintenance Menu 3-4
Menu Definition Menu 7-2
Table Definition Menu 12-5
Table Entry Definition Menu 12-14, 12-15, 12-16
Window Level Help Definition Menu 10-17
Mapping Services
Primary Menu 25-4

menus (*continued*)
MODS
Apply Definition Menu 17-33
Browse Definition Menu 17-37
Compare Definitions Menu 17-29
Copy Definition Menu 17-9
Definition Utility Menu 17-5
Delete Definition Menu 17-24
Library Definition Menu 16-6
Log Menu 17-16
Utility Menu 16-2
message definition 11-1
adding 11-3
browsing 11-9
copying 11-11
deleting 11-10
listing 11-7
updating 11-10
Message Explanation 11-4
message help 2-10, 11-4
Message ID 11-2, 11-3, 11-5
Message Prefix 4-4
Message Text 11-4
messages 2-10
Mlev, panel definition 6-8
modification level
panel definition 6-8
MODS
access authority 3-5
Apply Definition Menu 17-33
Apply File Definition panel 17-32
Browse Definition Menu 17-37
Browse File Definition panel 17-36
Compare Definitions Menu 17-29
Compare File Definitions panel 17-27
Component List
Application data 17-12
Common data 17-13
component report formats 15-2
component reports 15-1
component reports, printing 15-2
component reports, restricting contents of 15-3
control file 2-17
Control File Entity Apply panel 17-35
Control File Entity COMPARE panel 17-31
Control File Entity Copy panel 17-15
MODS (*continued*)
Control File Entity Delete panel 17-26
Control File Entity Move panel 17-22
Copy Definition Menu 17-9
Copy File Definition panel 17-8
Copy Panel 6-9
database 2-17
Definition Report Search panel
identifying test variables 15-5
using the Test fields 15-4
Definition Utility Menu 17-5
Delete Definition Menu 17-24
Delete File Definition panel 17-23
Edit Panel C-2
Library Definition Menu 16-6
Log Menu 17-16
Log Records panel 17-17
Move File Definition panel 17-19
overview 1-1
Panel Copy List 16-4
Panel Library List panel 6-5
Panel Library Maintenance Menu 16-2
Panel List, sorting 6-10
Search Definitions panel 17-38
Unformatted List of File Definition panel 17-40
Unformatted Records List 17-41
MOVE command, panel editor C-10
multiple select list 2-7

N

N command, text editor B-3
naming standards 5-1
NCL procedure $EDMENU 6-3
New Panel Name, panel definition 6-4
NEXT command, text editor B-8
NOTEPAD command 14-4
null fields, defining 6-20
NULLS command
panel editor C-10
text editor B-8
numbered list 2-7

O

O command
  panel editor C-14
  text editor B-3
Object Class Specification, accessing 3-4
Object Services Class ID validation table 12-8
options, menus, defining 7-6
OSATT
  edit type for validation tables 12-7
OSFLD
  edit type for validation tables 12-7
output-only fields, defining 6-20

P

PAGE command, panel editor C-11
panel
  preparsing 6-19
  processing options
    #OPT panel control statement 6-48
  queue 6-3
  skipping 2-6
  statistics 6-7
panel control statements 6-17
  #ALIAS 6-22
panel definition
  adding 6-6
  browsing 6-8, 6-10
  copying 6-9
    between libraries (on different paths) 16-3
    between libraries (on the same path) 6-13
    within a library 6-11
  deleting 6-8, 6-11
  designing
    data 6-18
    field characters 6-19
    field types 6-20
    fields 6-19
    function key area 6-26
    input padding and justification 6-25
    output padding and justification 6-22
    panel control statements 6-17
  editing 6-8, C-1
  information 6-8, 6-12
panel definition (*continued*)
  listing 6-7
  moving between libraries 6-13
  printing 6-8, 6-11
  renaming 6-9, 6-12
  selecting 6-8
  testing 6-8
  updating 6-10
  viewing in display format 6-11
panel definition field character, modifying 6-32
panel domain 2-8, 9-1
  definition 9-1
    browsing 9-7
    copying 9-8
    deleting 9-8
    listing 9-5
    updating 9-8
  reloading 9-9
panel editor
  block edit commands C-13
  BOTTOM command C-5
  CANCEL command C-5
  CAPS command C-5
  CHANGE command C-6
  commands summary C-1
  COPY command C-7
  CREATE command C-7
  DOWN command C-9
  FIND command C-9
  FSM command C-9
  HALF command C-9
  homing the cursor C-4
  LEFT command C-10
  line commands C-12
    '((ShiftLeft)' C-16
    ')(ShiftRight)' C-16
    A (After) C-13
    B (Before) C-13
    C (Copy) C-13
    COLS C-13
    D (Delete) C-14
    I (Insert) C-14
    IB (Insert Before) C-14
    M (Move) C-14
    O (Overlay) C-14
    Q (SAVE BLOCK) C-15
    R (Repeat) C-16
  MAX command C-10
  MOVE command C-10
  NULLS command C-10
  PAGE command C-11
  panel format C-4

panel editor (*continued*)
primary commands C-4
QFREE command C-11
REPLACE command C-11
RESET command C-11
RIGHT command C-11
SAVE command C-11
scrolling C-3
TOP command C-11
UP command C-12
panel library 6-2, 16-1
accessing 16-2
copying panels between libraries (any path) 16-3
defining 16-6
member 6-2
member statistics 6-7
panel queue 6-3
selection list 6-5
Panel Maintenance
accessing 3-3
Panel Name
element definition 9-14
panel definition 6-4
Panel Navigator
Path Definition Panel 9-19
panel path 2-8, 2-17, 6-2, 16-1
default path (PANELS) 6-2
panel path definition 9-2
adding 9-18
browsing 9-20
copying 9-20
deleting 9-20
listing 9-16
updating 9-20
Panel Services
#ALIAS panel control statement 6-27
#ERR panel control statement 6-29
#FLD panel control statement 6-32
#NOTE panel control statement 6-47
#OPT panel control statement 6-48
#TRAILER panel control statement 6-55
panel control statements
#ALIAS 6-27
#ERR 6-29
#FLD 6-32
#NOTE 6-47
#OPT 6-48
#TRAILER 6-55
panel skipping 2-6
panels 6-1
CAS
Application Definition List (page 1) 4-5
Application Definition List (page 2) 4-6
Application Level Help Definition Panel 10-5, 10-6
Criteria Definition List (page 1) 13-9
Criteria Definition List (page 2) 13-9
Criteria Definition Panel 13-5
Criteria Text Panel 13-7
List Definition List (page 1) 8-16
List Definition List (page 2) 8-16
List Definition List (page 4) 8-18
List Description Panel 8-6
List Format Panel 8-13
Menu Definition List Panel 7-9
Menu Description Panel 7-3
Message Definition List 11-8
Message Text/Explanation panel 11-3
Panel Domain Definition List (page1) 9-6
Panel Domain Definition panel 9-4
Panel Domain Element Definition List (page 1) 9-10
Panel Domain Element Definition List (page 3) 9-11
Panel Domain Element Definition Panel 9-13
Select Alias Help List 10-26
Select Alias Help Panel 10-25
Table Entry Definition List (page 1) 12-20
Table Entry Definition List (page 2) 12-21
Table Entry Definition Panel 12-17
maintaining 6-1

panels (*continued*)
CAS (*continued*)
Mapping Services
Map Definition panel 25-5
Map Definitions List (page 1) 25-7
Map Definitions List (page 2) 25-7
Map Definitions List (page 3) 25-8
Map Definitions List (page 4) 25-8, 25-10, 25-12
Map Definitions List (page 5) 25-12
MODS
Application ID List 17-11
Apply File Definition panel 17-32
Browse File Definition panel 17-36
Compare File Definitions panel 17-27
Component List
Application data 17-12
Common data 17-13
Control File Entity Apply panel 17-35
Control File Entity COMPARE panel 17-31
Control File Entity Copy panel 17-15
Control File Entity Delete panel 17-26
Control File Entity Move panel 17-22
Copy File Definition panel 17-8
Copy Panel 6-9
Delete File Definition panel 17-23
Edit Panel C-2
Log Records panel 17-17
Move File Definition panel 17-19
Panel Copy List 16-4
Panel Library List panel 6-5
Search Definitions panel 17-38
Unformatted List of File Definition panel 17-40
Unformatted Records List 17-41
Panel Navigator
Path Definition Panel 9-19
PSM
Confirm Printer panel 17-17
panels (*continued*)
searching for character strings 6-16
UAMS
Access Authorities panel 3-5
MODS Details panel 3-6
PASSWORD command 14-4
path 2-8, 2-17, 6-2, 16-1
definition 9-2
adding 9-18
listing 9-16
path name, panel definition 6-4
pick list 2-7
PQUEUE command 14-4
PREV command, text editor B-8
Prim and Sec Space control file 17-9
primary menu, defining 7-4
Printer Name
printing the log 17-18
printing a map definition 25-11
programming interface 2-5
prompted fields 2-11
prompting 2-11
PSKIP command 14-5
PSM : Confirm Printer panel 17-17
PUT tapes, applying 17-6

Q

Q command
panel editor C-15
text editor B-3
QFREE command, panel editor C-11

R

R command
panel editor C-16
text editor B-4
reloading
command definitions 14-9
panel domain 9-9
validation tables 12-2
REPLACE command, panel editor C-11
Replace Entity control file 17-10
Replace Like-Named Panels option 6-15, 16-3
Report Maintenance, accessing 3-4

Report Selection, accessing 3-5
Report Writer, overview 2-14
report, MODS components 15-1
formats 15-2
printing 15-2
restricting contents of 15-3
RESET command
panel editor C-11
text editor B-8
Reset control file maintenance 17-28
RETRIEVE command 14-5
return codes, $CACALL 18-4
reverse video help text 10-30
RIGHT command, panel editor C-11
Run Time Panel, criteria definition 13-5

S

SAVE command, panel editor C-11
scrolling amounts, panel editor C-3
searching character strings in panels 6-16
security, access to MODS 3-5
Select Mnemonic list definition 8-8
selector pen detectable fields 6-20
Sequence Number
validation table entry 12-18
Sequence numbers?
validation table 12-7
service procedure
list 8-7, 20-1
menu 2-6, 7-4, 19-1
shorthand time and date entry D-1
Shrvars menu definition 7-6
single select list 2-7
Sort Expression list definition 8-11
SPLIT command 14-5
Start Column, help file 10-17
START command 14-5
Start Row, help file 10-17
Status
element definition 9-13
path definition 9-20
Sterling Software
web site A-2
SWAP command 14-5
System Action, message definition 11-6

T

TABLE
edit type for validation tables 12-7
TE command, text editor B-4
text browse facility 2-13
Text Description
validation table definition 12-9
text editor 2-13
ALL command B-6
block commands B-1
CHANGE command B-7
commands B-1
FIND command B-7
FIRST command B-7
FLOW command B-7
FLOWM command B-7
LAST command B-7
line commands
(B-5
) B-5
A (After) B-1
B (Before) B-2
C (Copy) B-2
COLS B-2
D (Delete) B-2
I (Insert) B-2
LC (Lowercase) B-2
M (Move) B-2
N (Add to Notepad) B-3
O (Overlay) B-3
Q (SAVE BLOCK) B-3
QA B-3
QB B-3
R (Repeat) B-4
TE (Text Entry) B-4
TF (Text Flow) B-4
TFM (Text Flow to Margins) B-4
TS (Text Split) B-4
UC (Uppercase) B-4
MARGINS command B-8
NEXT command B-8
NULLS command B-8
PREV command B-8
RESET command B-8
Text Name, element definition 9-14
Text Title, element definition 9-14
TF command
text editor B-4
TFM command, text editor B-4
time, shorthand entry D-1

title line, menu 7-4
To Element, path definition 9-19
TOP command, panel editor C-11
top left corner display, menu 7-4
top right corner display, menu 7-4
trailer panel control statement 6-55
TS command, text editor B-4
tutorial, defining 10-12
Type, element definition 9-13

U

UAMS
    Access Authorities panel 3-5
    MODS Details panel 3-6
UC command, text editor B-4
underline, help text 10-30
underscore, help text 10-30
UP command, panel editor C-12
updating a map definition 25-10
User Action, message definition 11-6
user comments in panel definitions
    #NOTE panel control statement 6-47
user info box, displaying on menus 7-4
Userid
    criteria definition 13-5
Userid, criteria definition 13-4

V

Validation exit 12-7
Validation Manager 2-11
validation table
    adding 12-6
    browsing 12-9
    copying 12-10
    copying (table plus entries) 12-15
    deleting 12-10
    deleting (table plus entries) 12-15
    listing 12-10
    reloading 12-2
    updating 12-10
validation table entry
    adding 12-17
    browsing 12-19
    copying 12-19
    deleting 12-19
    listing 12-20
    updating 12-19
VALIGN operand 6-24
variables in help text 10-30
viewing a map structure 25-12
Volume, control file allocation 17-9

W

Web Site, NMD A-2
Weight, path definition 9-19
WHERE command 14-5
Window End Column, help file 10-9
Window End Row, help file 10-8
Window Level Help File
    adding 10-18
    browsing 10-19
    copying 10-19
    deleting 10-19
    listing 10-20
    updating 10-19
    viewing 10-20
Window Start Column, help file 10-8
Window Start Row, help file 10-8